VYSOKÁ ŠKOLA BÁŇSKÁ – TECHNICKÁ UNIVERZITA OSTRAVA

Laboratorní stav pro ověřování funkce ABS jednotky

Laboratory Stand for Testing Operation of the ABS Unit

Vedúci diplomovej práce: doc. Ing. Kočí Petr, Ph.D.
Poslucháč: Patrik Urban
Dátum odovzdania práce: 15. 5. 2013

Poďakovanie

Ďakujem doc. Ing. Kočímu Petrovi, Ph.D. za odborné a metodické vedenie, ktoré mi poskytol pri vypracovávaní diplomovej práce, Vladimírovi Starému za prevedené konštrukčné úpravy na modeli, Ing. Marekovi Babiuchovi, Ph.D. a doc. Ing Lenke Landryovej, CSc. za korekčné textové úpravy.

Prehlásenie študenta

Prehlasujem, že som celú diplomovú prácu vrátané príloh vypracoval samostatne pod vedením vedúceho diplomovej práce a uviedol som všetky použité podklady a literatúru.

V Ostrave :..... ….....

Celé meno študenta

Prehlasujem, že

- bol oboznámený s tým, že na moju diplomovú (bakalársku) prácu sa plne vzťahuje zákon č. 121/2000 Zb. – autorský zákon, najmä §35 – použitie práce v rámci občianskych a náboženských obradov v rámci školských predstavení a použite ako školskej práce a §60 – školská práca.
- beriem na vedomie že Vysoká škola báňská – Technická univerzita Ostrava (ďalej len VŠB-TUO) má právo neziskovo, ku svojej vnútornej potrebe diplomovú (bakalársku) prácu použiť(§35 odst. 3).
- Súhlasím s tým, že jeden výtlačok diplomovej (bakalárskej) práce bude uložený v Ústrednej knižnici VŠB-TUO k prezenčnému nahliadnutiu a jeden výtlačok bude uložený u vedúceho diplomovej (bakalárskej) práce. Súhlasím s tým, že údaje o diplomovej (bakalárskej) práci budú zverejnené v informačnom systéme VŠB-TUO.
- Bolo dohodnuté, že s VŠB-TUO, v prípade záujmu z jej strany, uzavriem licenčnú zmluvu s oprávnením používať prácu v rozsahu §12 odst. 4 autorského zákona.
- Bolo dohodnuté, že použiť svoju prácu – diplomovú (bakalársku) prácu, alebo poskytnúť licenciu k jej využitiu môžem len so súhlasom VŠB-TUO, ktorá je oprávnená v takomto prípade odo mňa požadovať primeraný príspevok na úhradu nákladov, ktoré boli VŠB-TUO na vytvorenie práce vynaložené(až do plnej výšky).
- Beriem na vedomie, že odovzdaním svojej práce súhlasím so zverejnením svojej práce podľa zákona č. 111/1998 Zb., o vysokých školách a o zmene a doplnení ďalších zákonov(zákon o vysokých školách), v znení neskorších predpisov bez ohľadu na výsledok jej obhajoby.

V Ostrave:.....

Patrik Urban
Dolné Hámre 282
96661, Hodruša - Hámre

Anotácie diplomovej práce

Urban, P.: Laboratorní stav pro ověřování funkce ABS jednotky: kat. ATŘ – 352 VŠB-TUO, 2011. 57 s. Bakalárska práca, vedúci: doc. Ing. Kočí Petr, Ph.D.

Diplomová práca sa zaoberá navrhnutím laboratórneho standu pre testovanie ABS jednotiek. ABS jednotka je vytvorená z jednoprocesorovej dosky, snímačov z Hallových sond a magnetických koliesok. Akčné zásahy sú riadené cez RC servomechanizmy. Princíp modelu vyplýva z brzdenia hmoty tvorenej zotrvačníkom simulujúcej vozovku pomocou kolesa brzdeného mechanickou kotúčovou brzdou tak, aby nastal minimálny sklz. Po rozbehu modelu má za úlohu nasadený algoritmus začať brzdiť s mechanickou brzdou, pri deji brzdenia snímať otáčky zotrvačníku, kolesa a podľa nameraných dát určiť sklz a tak vytvoriť akčný zásah na brzdu. Namerané hodnoty je možné vizualizovať pomocou vytvorenej aplikácie pre export nameraných hodnôt. Práca má slúžiť pre návrh algoritmov schopných pracovať zo signálmi senzorov a následne riešiť akčné zásahy brzdy.

Annotation of diploma thesis

Urban, P.: Laboratory Stand for Testing Operation of the ABS Unit. Ostrava: Dept. Of Control Systems and Instrumentation – 352 VŠB-TUO, 2011. 57 p. Master Thesis, Supervisor: doc. Ing. Kočí Petr, Ph.D.

My thesis is focused on designing laboratory stand for testing ABS units. The ABS unit is composed of a uniprocessor board, sensors from Hall probes and magnetic wheels. Interventions are controlled by the RC servomechanisms. Idea of the model is in stopping a mass created by a flywheel, which represents roadway by wheel braked with mechanical disc brake in minimal skid way. After model accelerates, used algorithm starts breaking with mechanical brake and simultaneously read operating speed of the flywheel. According to the measured data, algorithm identifies skid and thus creates intervention on the brake. Measured values can be visualized by the application created for export of the measured values. Thesis serves for designing algorithms able to work with sensor signals and afterwards solve brakes interventions.

Obsah

Zoznam použitých skratiek a symbolov

a x b	dĺžka x šírka
Cor	statické zaťaženie
Cr	dynamické zaťaženie
D	priemer otvoru ložiska
e	vzdialenosť k stredu upínacím skrutkám
F	sila
g	gravitačné zrýchlenie
H	celková výška ložiska
h	vzdialenosť k stredu ložiska od spodnej hrany
$\mathbf{m_c}$, $\mathbf{m_h}$, $\mathbf{m_z}$	celková hmotnosť, hmotnosť hriadele, hmotnosť zotrvačníka
RA, RB	reakcie v ložiskách
$\mathbf{r_z}$, $\mathbf{r_h}$	polomer zotrvačníka, polomer hriadeľa
v	rýchlosť
$\mathbf{v_z}$, $\mathbf{v_h}$	výška zotrvačníka, výška hriadeľa
μ	koeficient trenia

Úvod

Cieľom diplomovej práce je preštudovanie už navrhnutého ideového konštrukčného modelu z ktorého ďalej bude vychádzať praktický výučbový model ABS. Potrebným štúdiom modelu je treba definovať model o skutočné rozmery, podľa ktorých bude daný model postavený.

Vytvorený model navrhnutý programom PRO/ENGINEER možno rozložiť na dve základné oblasti, strojársku a elektronickú.

Strojárska časť pozostáva z hlavnej pevnej konštrukcie, na ktorú sú podľa vytvoreného modelu a vlastností potrebných pre simuláciu umiestňované komponenty. Konštrukcia je tak doplnená o potrebné držiaky pre motor, brzdu s kotúčom, domčeky s hriadeľmi, kolesami a zotrvačníkom.

Ovládanie je podľa vybratej jednoprocesorovej dosky a pridaním jednotlivých akčných členov a senzorov v podobe servomechanizmov a Hallových sónd.

Definíciou strojárskej a elektrotechnickej časti je možné pristúpiť k návrhu a testu algoritmov, s ktorými bude model skúšaný. Porovnávané budú hlavne reakčné časy a správania sa pri zmene jednotlivých algoritmov. Pri ich návrhu je využívaná napísaná aplikácia pre vizualizáciu a export dát. S vytvorením aplikácie sa zjednodušuje ukážková názornosť.

Testovanie je vytvárané za použitia simulačného modelu, pri ktorom bude skúmaný rozdiel medzi simulácia programom pre brzdenie s odôvodnením a porovnaním nameraných údajov z jednotlivých algoritmov pre ABS s možnosťou, kedy nastáva brzdenie šmykom.

Podľa hodnôt získaných v jednotlivých priebehoch je vyvodený záver a je stanovené možné riešenia do budúcna nesúce zlepšenie daných charakteristík.

1 Systém ABS

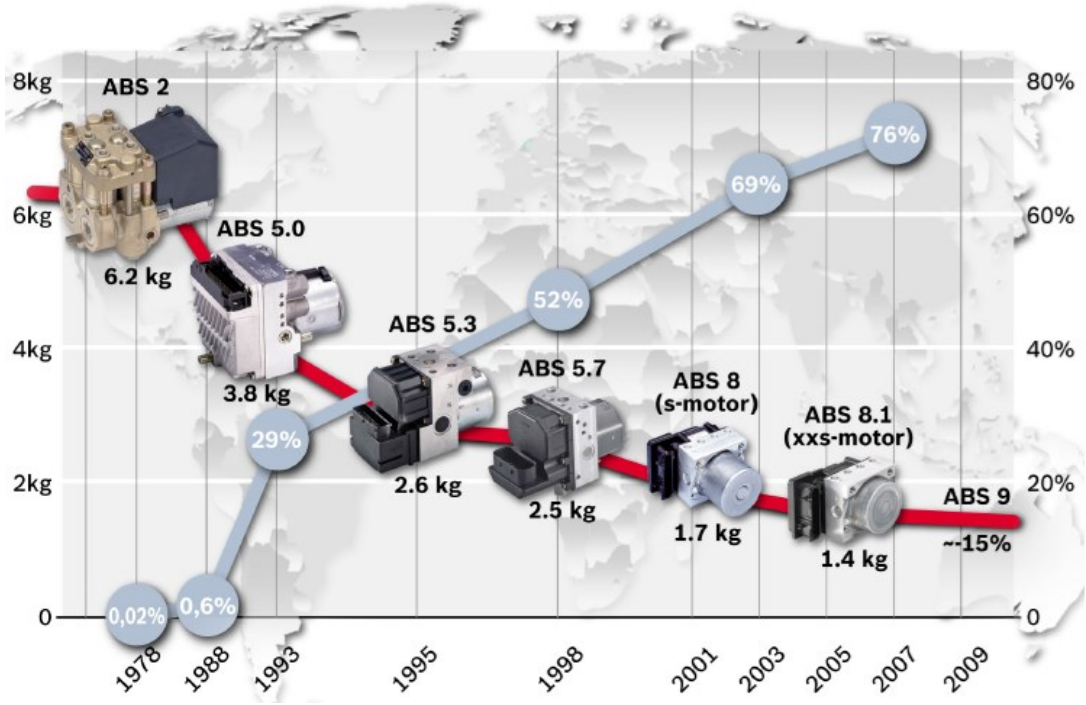
Antiblokovací systém je zo skupiny aktívnych prvkov najznámejší. Prvý pokus o principiálny model ABS predviedol v roku 1908 J.E. Francis pre lokomotívy. Najviac vývoja v tomto segmente robí firma Bosch, ktorá si nechala zariadenie v roku 1936 patentovať. Nasadené v automobiloch bolo roku 1978 firmou Mercedes. Dnes je systém ako základná a povinná výbava automobilov, príplatková výbava u motocyklov [http://www.isa.own.cz/,2010].

1.1. Vývoj jednotiek ABS

Od uvedenia prvého systému označovaného 2S uplynulo viac ako 30 rokov. Pri vývoji systému prišlo:

- k redukcii použitých súčiastok
- zjednodušenie systému
- spojenie hydraulickej a elektronickej jednotky
- zníženie hmotnosti

Dnes je v automobiloch používaný systém ABS 8.1, ktorý prináša zmeny



Obrázok 1.1 Vývoj riadiacich jednotiek ABS [Cappa,2008]

regulácii v spätnej dodávky počas danej situácie brzdenia. Zmena dovoluje použiť menší elektromotor a redukciu konštrukčnej velkosti. Agregát je vhodný pre

použitie u objemov 1 liter. Elektromotor jednotky ponúka variabilné riadenie otáčok, čím potlačuje nežiaduce vibrácie a hluky. Zmenšená energetická náročnosť značne šetrí elektrickú sieť automobilu.

Najnovší systém prevažne nasadený v motocykloch je označovaný ABS 9.0. Oproti predchodcovi je celý systém o polovicu menší a sú na trhu jeho tri verzie:

- **base** – verzia vhodná pre neskúsených vodičov, nerobí mu problém ani náhle zmeny povrchu.
- **plus** - vhodná pre motocykle s vačším výkonom, verzia je doplnená ďalším tlakovým senzorom, ktorý sa uplatní pri núdzovom brzdení a zabráni tým prevráteniu motocykla.
- **enhanced** – obsahuje naviac funkciu eCBS, podľa tvrdení výrobcu pri brzdení stačí jazdcovi stlačiť len jednu brzdu a systém automaticky aktivuje ďalšiu, bez toho aby jazdec musel zvýšiť tlak.

1.2. Regulácia a možnosti jej zapojenia

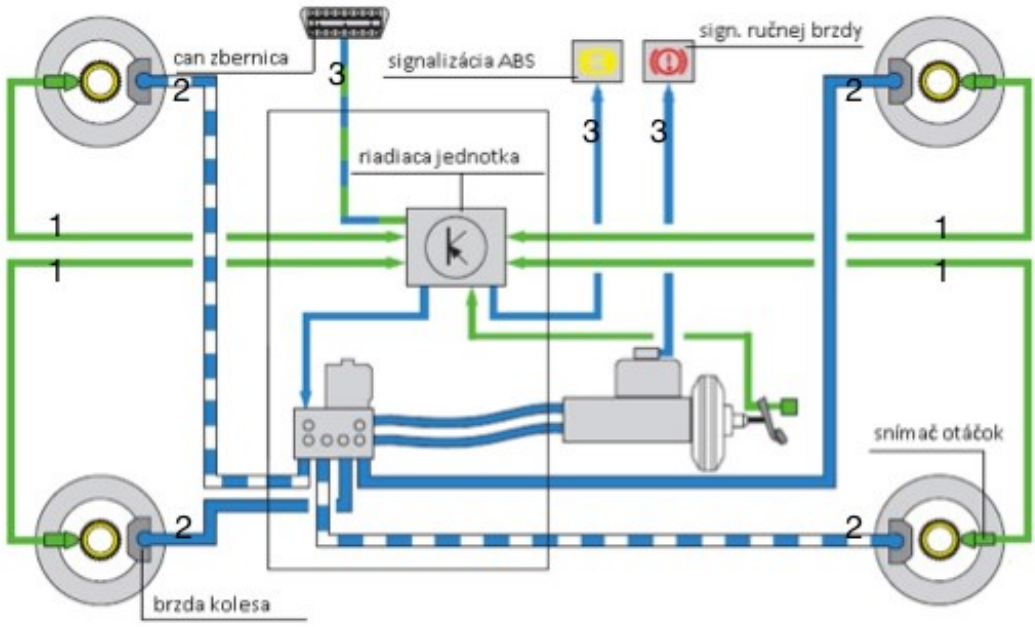
Úlohou je pri brzdení zabezpečiť najkratšiu brzdnú dráhu a stabilitu vozidla, podľa umiestnenia snímačov a spracovania môžeme systém rozdeliť do troch skupín:

- **individuálna regulácia –** každé koleso obsahuje samostatný snímač a solenoidný ventil, brzdná dráha je najkratšia zo všetkých používaných typov, pri rozdielnej adhézii dochádza k točivému momentu vozidla a nie je zaručená dostatočne dobrá smerová stabilita.
- **zmiešaná regulácia –** Brzdy sú zapojené diagonálne. Predná náprava obsahuje individuálnu reguláciu a zadná náprava funguje na metóde „Select Low“. Metóda „Select Low“ vyberá potrebné hodnoty z kolesa s menším súčiniteľom trenia.
- **modifikovaná regulácia –** Regulácia je upravená pre prednú nápravu. Zadnú nápravu reguluje individuálna regulácia. Zmena na prednej náprave vychádza z princíp „Select Low“ so zmenou, že pri blokovaní kolesa ostáva tlak v neblokujúcom kolese na konštantnej hodnote. Tlak v blokujúcom kolese je znižovaný až kým koleso nedosiahne referenčnú rýchlosť.

1.3. Činnosť regulácie

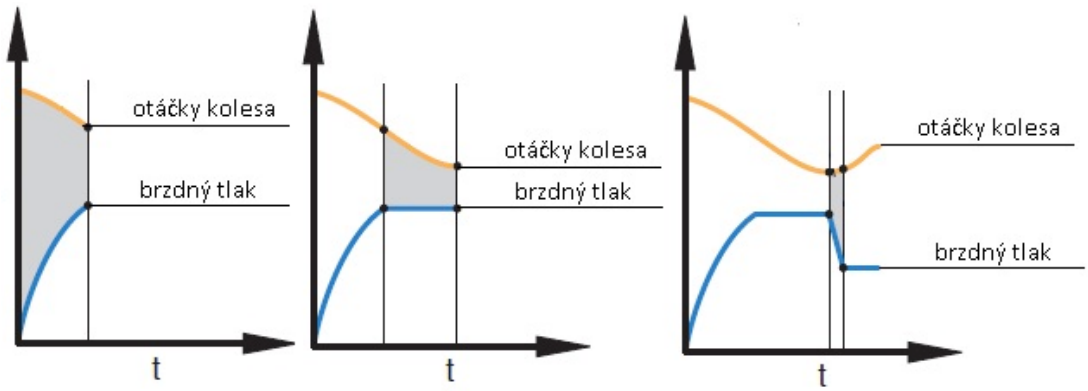
Činnosť môžeme rozdeliť na dve časti. Elektronická časť systému neustále monitoruje a porovnáva hodnoty signálov pracujúcich na princípe elektromagnetickej

indukcie alebo Hallovho javu. Zo signálov zisťuje aktuálnu rýchlosť a porovnáva ju voči referenčnej rýchlosti vozidla. Referenčná rýchlosť je získavaná z kolies uložených diagonálne, tým určí zrýchlenie, spomalenie, sklz kolesa.



Obrázok 1.2 Principiálna schéma zapojenia systému ABS - (1-zisk nameraných hodnôt,2-ovládanie hydraulických povelov, 3 - pripojenie externých zariadení (diagnostika, signálne kontrolky)) [zdroj: Škoda auto a.s.]

Pokiaľ elektronická časť zistí, že dochádza k blokovaniu kolesa, aktivuje v hydraulickej časti príslušné elektromagnetické ventily. Regulácia tlaku vo ventiloch môže nadobúdať 3 stavy:

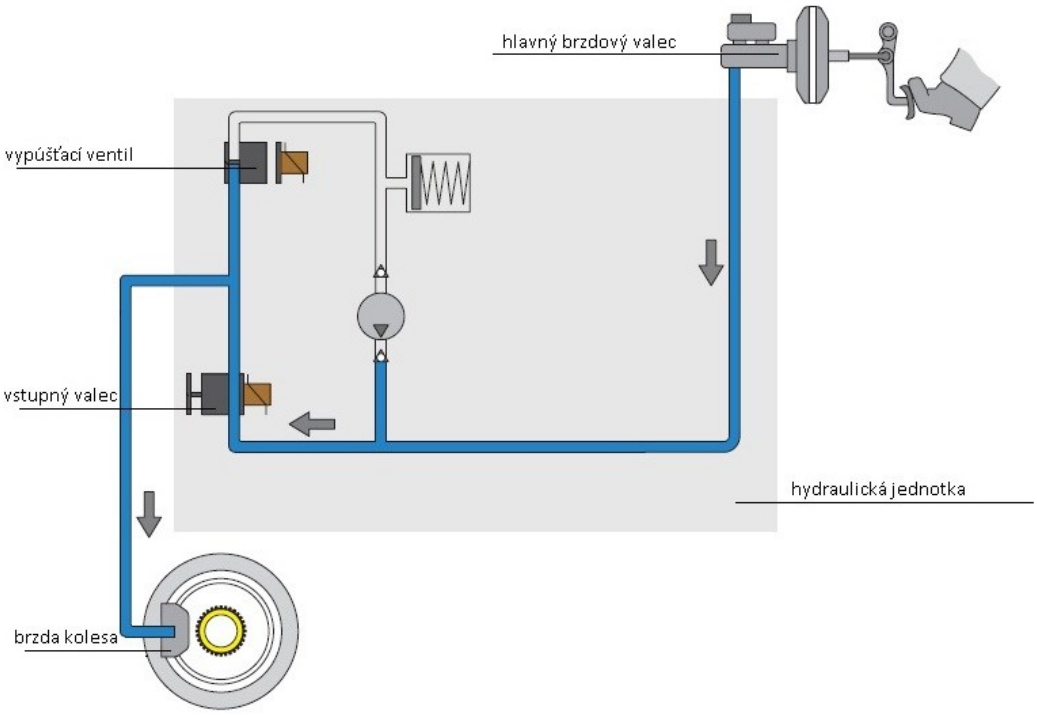


Obrázok 1.3 Simulácia brzdného tlaku –zvyšovanie - konštantný tlak - znižovanie tlaku [Škoda auto a.s., 2013]

- Stlačením brzdového pedála sa v hlavnom brzdovom valci vytvorí tlak potrebný k brzdeniu.

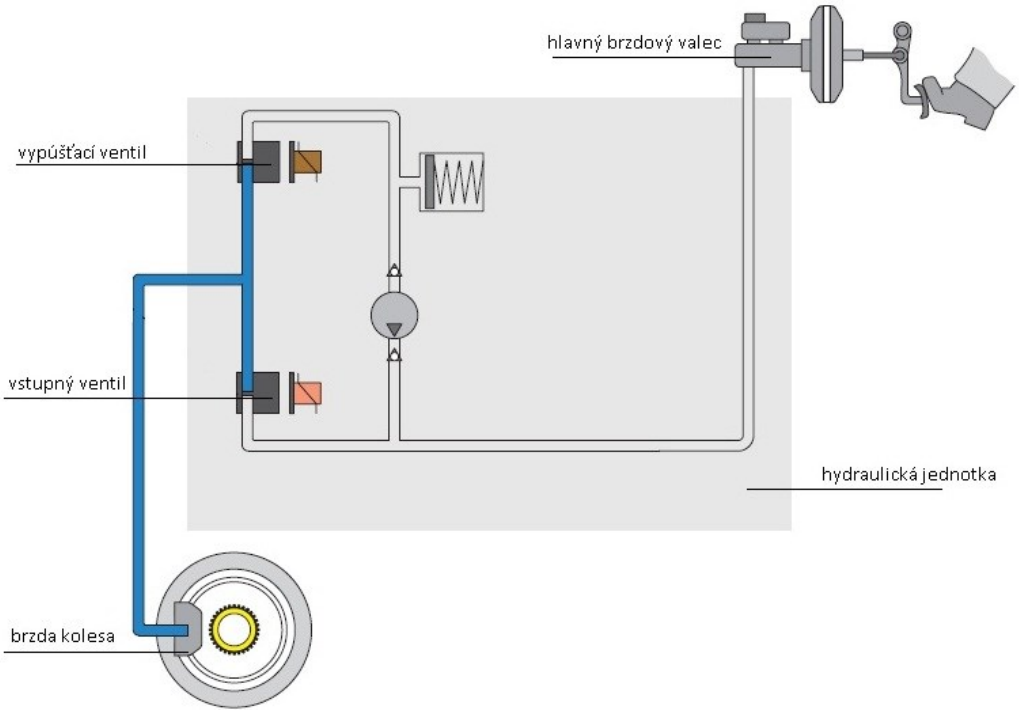
  Tlak je k brzdovým valcom prenášaný cez vstupný ventil, ktorý sa

nachádza v nevybudenom stave bez napätia tak isto, ako aj vypúšťací ventil.



Obrázok 1.4 Principiálna schéma zvyšovania tlaku [Škoda auto a.s., 2013]

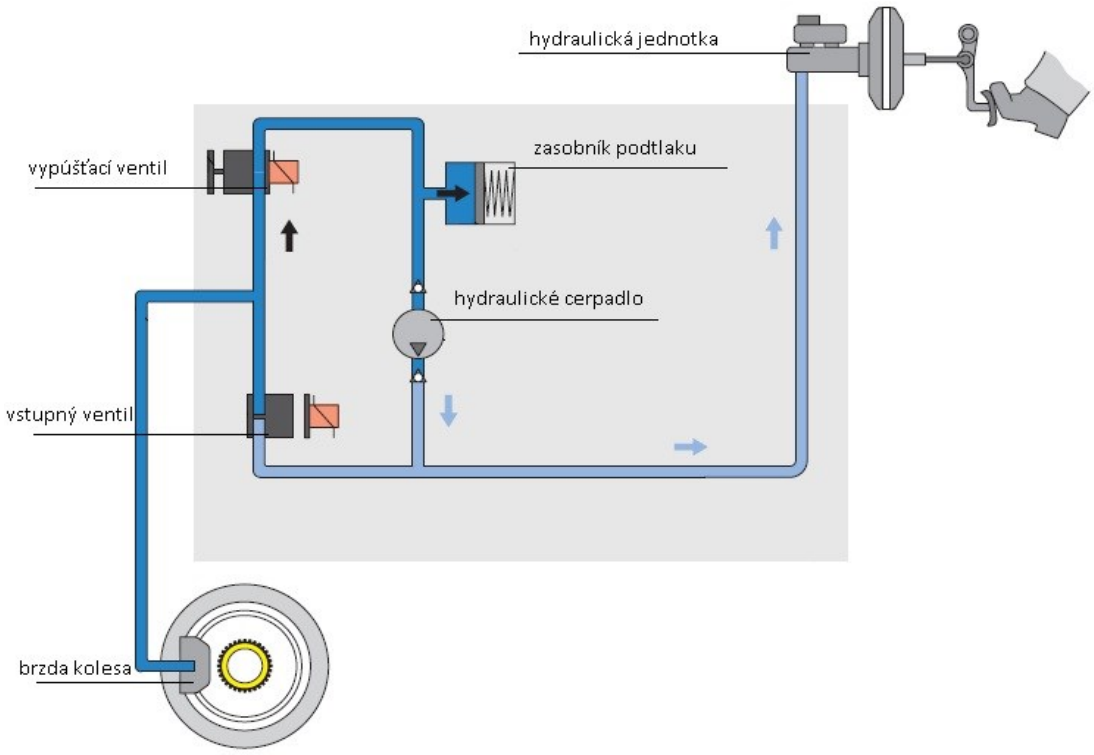
- V prípade, keď nastane stav blokovania kolesa, zasiahne do brzdenia jednotka ABS a zabráni ďalšiemu zvyšovaniu tlaku. Vstupný ventil zmení svoj stav



Obrázok 1.5 Principiálna schéma konštantného tlaku [Škoda auto a.s., 2013]

privedením vstupného napätia. Vypúšťací ventil je aj naďalej uzatvorený (bez napätia).

- V stave, v ktorom pretrváva blokovanie kolesa aj pri konštantnom brzdiacom tlaku, musí byť brzdový tlak znížený. Vypúšťací ventil je otvorený pomocou privedeného napätia a pomocou zásobníku podtlaku sa tlak v brzdovom systéme zníži. Hydraulické čerpadlo privedie brzdovú kvapalinu z podtlaku do hlavného brzdového valca. Brzdový pedál sa pri tejto situácii pohybuje smerom hore. Napätia na vstupnom ventile ponecháva ventil v uzavretom stave.



Obrázok 1.6 Principiálna schéma znižovania tlaku [Škoda auto a.s., 2013]

Tento cyklus je podľa potreby opakovaný 4 až 10 krát za sekundu. Po znížení rýchlosti pod hranicu 4km/h sa systém neaktivuje.

1.4. Požiadavky na systém ABS

Na systém sú kladené tie najdôležitejšie požiadavky, ktoré sú prienikom bezpečnosti, kvalitu systému, spoľahlivosť. Medzi najdôležitejšie patrí hlavne:

- Systém pri brzdení musí zabezpečiť stabilitu a riaditeľnosť automobilu pri akýchkoľvek podmienkach jazdnej dráhy. Pri brzdení má stabilita a riaditeľnosť prednosť pred skrátením dráhy.
- Regulácia musí reagovať až po minimálnu rýchlosť a okamžite sa prispôsobiť zmenám priľnavosti vozovky.
- Keď vozidlo brzdí v zákrute, musí ostať riaditeľné a stabilné s najkratšou brzdnou dráhou.
- Musí rozpoznať aquaplaning a reagovať naň.
- Systém sa pri zistení chyby vedúcej k ovplyvneniu brzdenia ABS vypne a vodiča informuje kontrolkou. [Inteligentné snímače v automobiloch, 2013]

2 Laboratórny stand pre overenie činnosti ABS jednotky

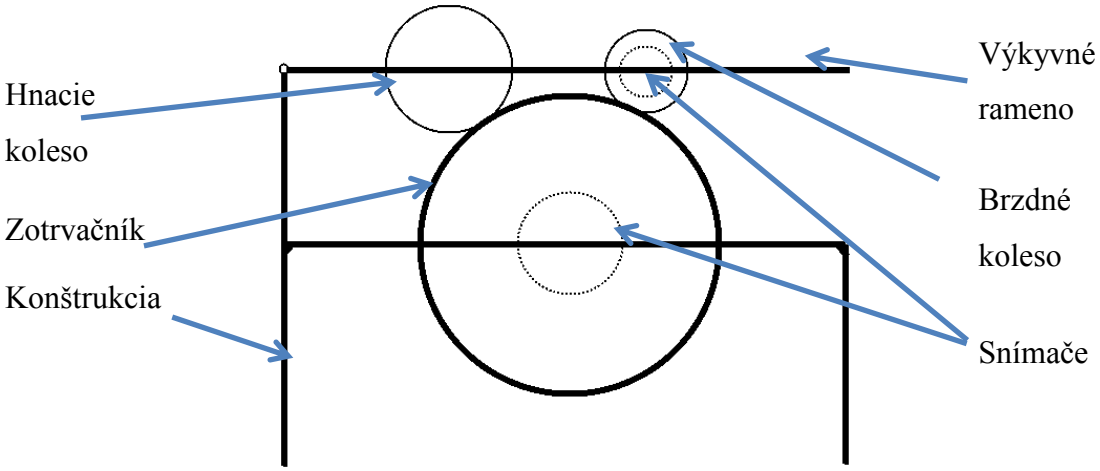
Riešenie modelu je navrhnuté v programe PRO/ENGINEER, čím bolo možné pri návrhu vytvoriť to najvhodnejšie rozloženie a usporiadanie daných komponentov nachádzajúcich sa na modeli.

Z modelu sú vypúšťané požiadavky: smerová stabilita, rozpoznanie aquaplaningu, toto však kladie dôraz hlavne na:

1. 4 – 10 zovretí / rozovretí čelustí
2. Zmenu brzdného ústrojenstva
3. Jednoduchosť

Schematickú štruktúru si môžeme znázorniť na obrázku 2.1, kde je schematicky znázornené umiestnenie súčiastok na modeli. Model môžeme rozdeliť do nasledujúcich blokov:

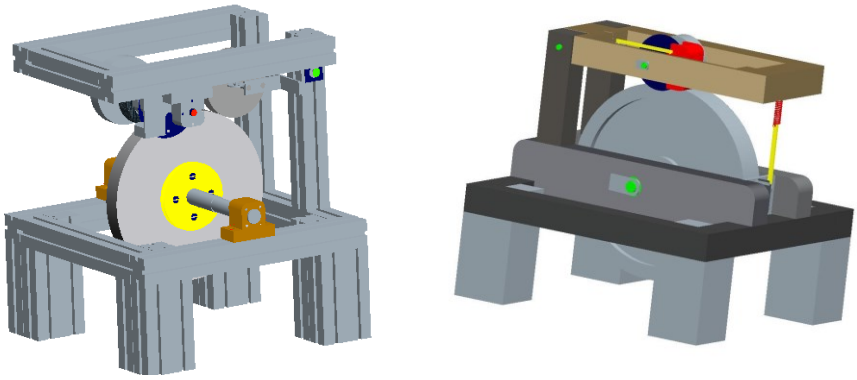
1. Podstava so zotrvačníkom
2. Výkyvné rameno s kolesom a brzdou
3. Senzory a akčné členy



Obrázok 2.1 Schematické znázornenie konštrukčného modelu

Model môžeme porovnať medzi ideou, z ktorej vychádza celý nápad a vytvorenou reálnou konštrukčnou časťou. Zmenu vidíme v odstránení prítlaku medzi pohyblivý ramenom a zotrvačníkom. Prítlak bol premiestený k motoru, a tým podľa zvoleného algoritmu vieme ovplyvniť referenčnú rýchlosť.

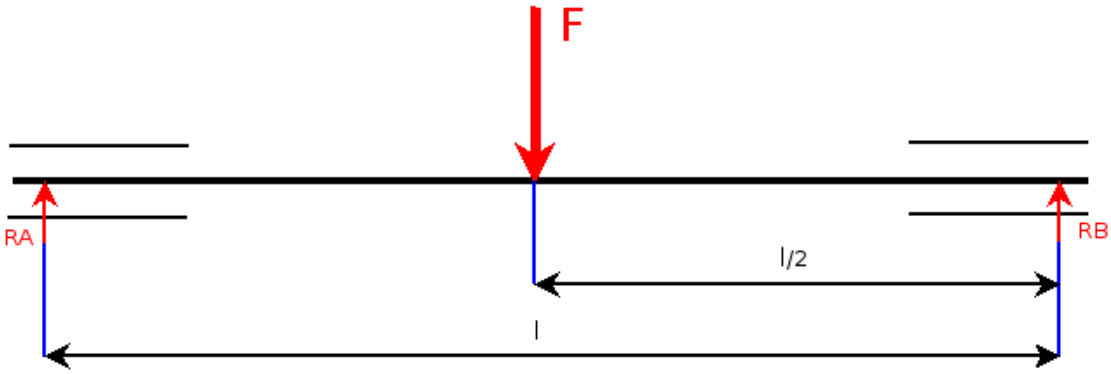
Roztočenie zotrvačníku je zmenené z ručného spôsobu, ktoré je uvedené v ideovom riešení na roztočenie pomocou motoru.



Obrázok 2.2 Vytvorený model(vľavo) – Ideový model (vpravo)

2.1. Podstava so zotrvačníkom

Celý model je tvorený hliníkovými profilmi s rozmermi 40x40 [mm], ktoré dodáva firma Habekorn Ulmer. Profily boli na mieru zhotovené podľa podkladov zostrojených v programe PRO/ENGINNER a okótovaných v AUTOCADE. Telo celého modelu tvorí podstava, ktorá je zároveň nosnou časťou pre zotrvačník a výkyvné rameno. Zotrvačník je umiestený na hriadeli, ktorá je ukotvená v dvoch ložiskách. Pre výpočet síl pôsobiacich v ložiskách použijeme obrázok 2.3.



Obrázok 2.3 Rozloženie sily pre hriadeľ zotrvačníka

Hriaďel so zotrvačníkom, ktorý je uložený v ložiskách, má hmotnosť,

$$m_c = m_z + m_{\square} = \pi r_z^2 v_z + \pi r_{\square}^2 v_{\square} = \pi(0{,}130^2 . 0{,}02 + 0{,}012^2 . 0{,}340) = \mathbf{9,52}[\square\square] \quad (1)$$

pôsobí rovnomerne silou

$$F = m_c g = \mathbf{93.39}\ [\mathbf{N}] \quad (2)$$

a na jednotlivé ložiská rovnomerne silou

$$RA = RB = \frac{F\frac{l}{2}}{l} = \frac{93,39.0,170}{0,340} = \mathbf{46,70}\ [\mathbf{N}] \quad (3)$$

Najvhodnejším výberom na osadenie a vhodnú manipuláciu bolo použitie jednoradových guľkových ložísk výklopných ložisiek umiestnených v domčekoch. Vybraté ložisko má nasledujúce parametre uvedené v tabuľke 2.1

Tabuľka 2-1 Parametre ložisiek zotrvačníka (d-priemer otvoru ložiska, e – vzdialenosť k stredu upínacím skrutkám, h – vzdialenosť k stredu ložiska od spodnej hrany, H – celková výška ložiska, a x b – dĺžka x šírka, Cor - statické zaťaženie, Cr - dynamické zaťaženie)

Typ	d [mm]	e [mm]	h [mm]	H[mm]	a x b[mm]	Cor[N]	Cr[N]
UCP 204	20	95	33,3	65	127 . 38	6650	12800

Zotrvačník pri konštantných otáčkach prepočítaných z obvodov medzi hnacím kolesom a zotrvačníkom dokáže uchovať energiu,

$$E_k = \frac{1}{2} mr^2\omega^2 = 0,5.8,32.0,130^2.8,04^2 = \mathbf{4,54}\ [\mathbf{kg.m^2}] \quad (4)$$

a pre jeho rozbeh je potrebné prekonať moment,

$$\varepsilon = \frac{a}{r} = \frac{\frac{\omega r}{t}}{r} = \frac{\frac{20,93.0,130}{2}}{0,130} = \mathbf{10,667}\ [-] \quad (5)$$

$$Mz = I\varepsilon = \frac{1}{2}mr^2\varepsilon = \frac{1}{2}.8,32.0,13^2.10,667 = \mathbf{0,707}\ [\mathbf{Nm}] \quad (6)$$

ktorý je možné na navrhnutom modeli prekonať dvoma možnými variantmi rozbehu:

- je ho možné roztočiť ručne pomocou roztáčacieho remeňa, toto roztočenie nemá dostatočnú dynamiku pre možné brzdenie
- druhou možnosťou je uviesť ho do pohybu pomocou motora.

Pohon je podľa vypočítaného momentu dimenzovaný trojnásobne z dôvodu prekonania odporov spôsobených odporom ložísk, trením brzdy. Jedná sa o motor, ktorý pohybuje strešným oknom na automobiloch. Podľa alternatívnych motorov má motor krútiaci moment väčší ako 3.5 Nm a experimentom zistené otáčky 200 ot/min. Sila, ktorou je roztáčaný zotrvačník, je prenášaná cez gumové koleso s priemerom 100 mm, a teda k zotrvačníku pomerom 2.6:1.[Alibaba.com,2013]

Pre možné zastavenie zotrvačníka je motor ovládaný servomechanizmom, ktorý vie medzi zotrvačníkom a hnacím kolesom meniť sklz a tým výslednú hnaciu silu.

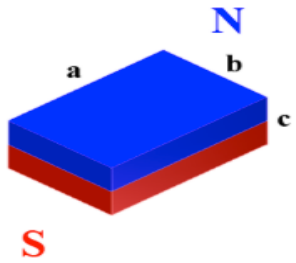
Regulovanie sklzu je zabezpečované cez tiahlo, ktoré ovláda digitálny servomechanizmus HK-752MG s nasledujúcimi potrebnými parametrami uvedenými

v tabuľke 2-2.[AIRHELI_SHOP,2013]

Tabuľka 2-2 Parametre servomechanizmu

Názov servomechanizmu	HK-752MG
Rýchlosť	0,13-0,10s/60° [4,8V/6V]
Ťah	5,1-6,3 kg/cm [4,8V/6V]
Rozmery	35 . 15 . 29,2 [mm]
Hmotnosť	28g

Snímanie otáčok je riešené pomocou magnetov snímaných Hallovou sondou. Na jednu otáčku pripadne 32 magnetov. Magnety sú nalepené a umiestené na pravej strane zotrvačníku, nalepené na obvode kružnice vytvorenej pomocou sústruhu. Kotúč je prichytený o roztáčací mechanizmus.



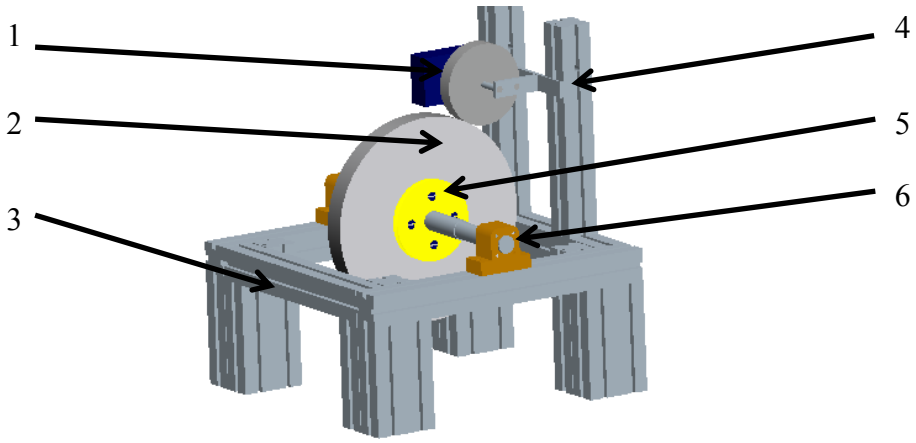
Obrázok 2.4 Magnet pre Hallovu sondu [Neomag,2012]

Hallova sonda ma nasledujúce parametre :

Tabuľka 2-3 Parametre Hallovej sondy

Napájacie napätie	max 18V
Odberaný prúd	max 8 mA
Výstupný prúd	max 20 mA

Celkové zobrazenie zotrvačníka s motorom pritláčaným servomechanizmom a umiestneným Hallovým senzorom vytvoreným v programe PRO/ENGINEER je na obrázku 2.5.



Obrázok 2.5 Podstava s motorom

Tabuľka 2-4 Popis prvkov na modeli

Číslo	*Popis*
1	*Motor s poháňacím kolesom*
2	*Zotrvačník*
3	*Základňa*
4	*Držiak ramena a motora*
5	*Kotúč so senzorami*
6	Hriadeľ s domčekmi

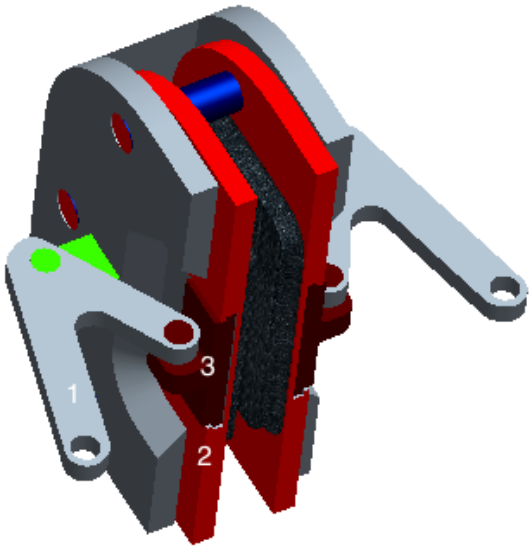
2.2. Výkyvné rameno

Zotrvačník umiestnený na hriadeli je potrebné brzdiť. Koleso, ktoré brzdí zotrvačník, je umiestené priamo nad ním. Rameno je pritláčané o zotrvačník silou :

Pri hladaní vhodného brzdiaceho mechanizmu boli v programe PRO/ENGINEER vytvorené v teoretickej rovine 3 a nájdené 2 už existujúce brzdy:

2.2.1. Kotúčová brzda - model

Model bol inšpirovaný skutočnou brzdou, ktorá sa nachádza v automobile. Pri modelovaní brzdy vzniklo množstvo konštrukčných spojení. Spojenia si môžeme všimnúť pri doštičke, ktorá je ukotvená v strmeni s brzdovou doštičkou. Pri tomto spojení sa jedná o vodiacu drážku mechanizmu. Pri tomto riešení sa jednalo o riešenie s najväčšími rozmermi. Prítlačné mechanizmy, ktoré sa nachádzajú na strmeni, by boli ovládané pomocou tiahla a servomechanizmu.

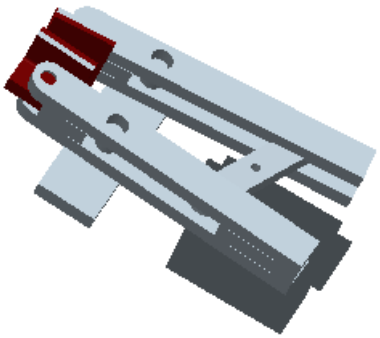


Obrázok 2.6 Strmeň kotúčovej brzdy (1 - Prítlačný mechanizmus, 2 - Doštička so strmeňom, 3 - Prítlačná doštička)

2.2.2. Brzda s excenterom

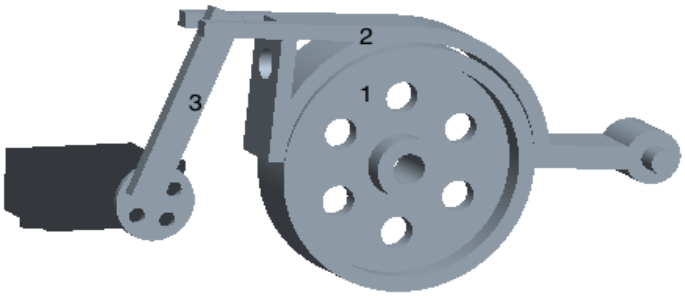
Návrh pozostáva zo servomechanizmu, ktorý unáša excenter.

Excenter roztláča výkyvné ramená umiestnené na držiaku. Ramená unášajú brzdové platničky, medzi ktorými je umiestnený brzdový kotúč. Na konci ramien musí byť v reálnom prevedení ešte pružina, ktorá by mechanizmus vracala do nulovej polohy. Mechanizmus zo všetkých navrhnutých mal najmenšie rozmery, aj najmenší počet spojení, ktoré by vadili rýchlosti ovládania. Na modeli by sa dalo riešenie do budúcna použiť.



Obrázok 2.7 Brzda s excentorom

2.2.3. Model pásovej brzdy

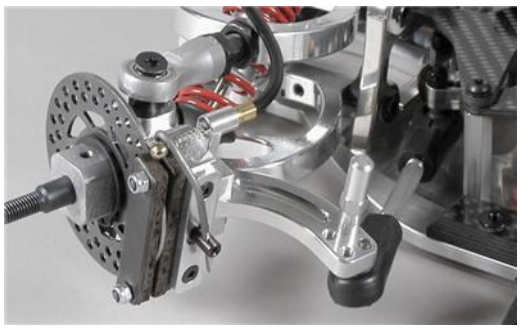


Obrázok 2.8 Pásová brzda (1- Brzdiace koliesko, 2- Prítlačný pás, 3 -Tiahlo)

Brzda vychádza z princípu klasickej pásovej brzdy. Brzdiace koliesko by bolo po obvode pribrzďované prítlačným pásom. Jedna strana obvodového pásu je ukotvená vo výkyvnom kĺbe a druhú stranu by pritláčalo servo pomocou tiahla. Na modeli by sa dalo riešenie do budúcna použiť.

2.2.4. RC Brzda

Pri hladaní možných riešení, ktoré už bol spomenuté, vychádzalo najlepšie model osadiť brzdou, ktorá sa nachádza na RC modeli spaľovacieho auta vo veľkosti 1:5. Brzda mala hlavné výhody, vyhotovenie bolo odskúšanie, nemôže nastať problém nefunkčnosti, je kompaktných rozmerov, dokáže ubrzdiť 5 - 7 kg.



Obrázok 2.9 Kotúčová brzda RC modelu [zdroj: https://www.fg-onlineshop.de,2012]

Kotúče by boli umiestnené po stranách ramena, kde by boli ovládane pomocou servomechanizmu.

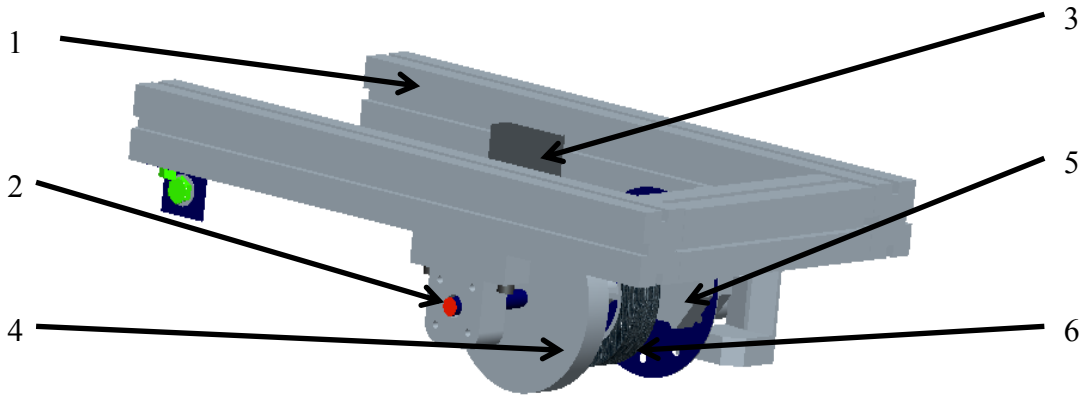
2.2.5. *Bicyklová brzda - finálna*

Na bicykloch poznáme dva typy bŕzd: mechanickú a hydraulickú. Na modeli je použitá mechanická.

Pri tejto voľbe je isté, že brzda dokáže vyvinúť dostatočne veľkú silu na to, aby koleso dostala do šmyku. Pri jej bežných použitiach na bicykloch musí ubrzdiť niekoľko násobne väčšie hmotnosti oproti tým, aké sú na modeli.

Obrázok 2.10 Kotúčová brzda z bicykla [zdroj: http://www.cykloabc.sk,2012]

Pre snímanie otáčok je použitý rovnaký spôsob, ako pri zotrvačníku pomocou Hallovej sondy, ovládanie brzdy je takisto pomocou servmochanizmu HK-752BG. Celkový pohľad na rameno vytvorené v programe PRO/ENGINEER máme na obrázku 2.11



Obrázok 2.11 Rameno ProEngineer

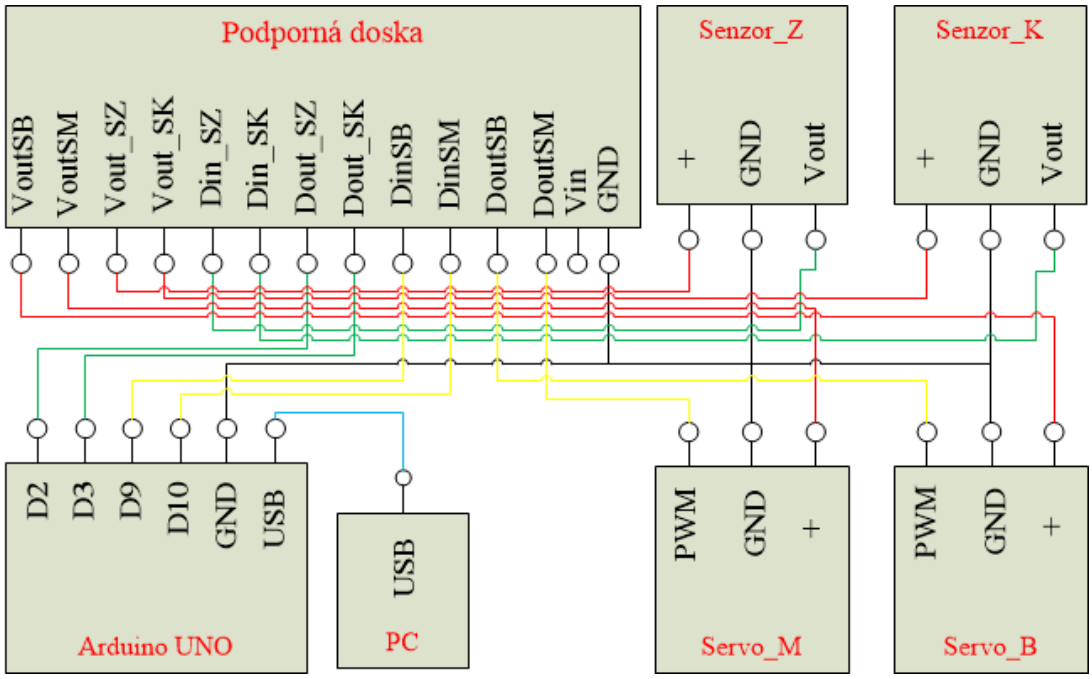
Tabuľka 2-5 Popis kompomentov na ramene

Číslo	***Popis***
1	*Rameno - šasi*
2	*Ložiská v domčekoch*
3	*Servomechanizmus*
4	*Senzor otáčok*
5	*Mechanická brzda s držiakom*
6	Koleso

2.3. Elektronika

Celé riadenie na modeli vychádza z blokového znázornenia na obrázku 2.11. Vizualizovanie a výber voľby simulácie vykonáva osobný počítač. Ten komunikuje s jednoprocesorovou doskou od Arduina. Táto doska riadi beh navrhnutého algoritmu pre simuláciu.

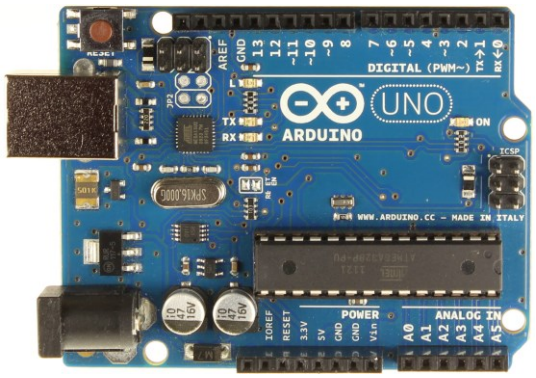
V podpornej doske je posilnené napájacie napätie pre senzory a servomechanizmy.



Obrázok 2.12 Bloková schéma elektronických častí (červená farba - napájanie, čierna - zem, žltá signálová časť pre servá, zelená - signálová časť pre senzory)

2.3.1. Vývojové nástroje

Pre vývoj na jednoprocesorových doskách existuje celá rada možností. Návrh je možný na vytvorených doskách s mikroprocesormi alebo použitých vývojových kitoch.



Obrázok 2.13 Arduino Uno

Najznámejšími predstaviteľmi v tejto oblasti sú Netduino, FEZ, GHI, využívajúci upravenú verziu Frameworku s názvom Microframework, pod ktorým je súčasť .Net Gadgeteer. Silou Gadgeteeru je možnosť prepojenia externých modulov v grafickom štúdio a tým

vytvorením potrebných inicializácií. Programovanie prebieha v Microsoft Visual C#. [NET Micro Framework a, 2012] [NET Micro Framework b,2012]

Použité Arduino ma vlastné programovacie prostredie s názvom Arduino IDE s multiplatformovovou inštaláciou.

Doska má osadený mikrokontrolér od firmy ATMEL ATMEGA328, ktorý pomocou rozhrania UART, komunikuje s počítačom. Používaný programovací jazyk Wiriing vychádza z jazyka C/C++ a obsahu množstvo knihovien pre obsluhu I/O portov. Výhodou je program, ktorý je uložený v pamäti a nepotrebuje externé spúštanie, preto môže doska byť použitá bez PC. Zvolená doska má špecifikáciu, viď. Tabuľka 2-6:

Tabuľka 2-6 Parametre ARDUINO UNO

Arduino UNO	
Takt procesora:	16 MHz
Flash pamäť:	32kB
SRAM pamäť:	2kB
EEPROM pamäť	1KB
Digitálne vstupy	14
Analógové vstupy	6

2.3.2. RC servomechanizmus

Servomechanizmy sú zariadenia, charakterizované nepatrnou vstupnou silou pre výstupnú zmenu polohy, pre ktorú je charakterizovaná veľká záťaž. Činnosťou mechanizmu je rýchle a presné sledovanie zmien vstupnej veličiny, na základe tohto sledovania pák upraviť veličinu výstupnú. Rozvoj servomechanizmov nastal vďaka vojenskej, leteckej doprave.[Servomechanizmy,2013]

Obrázok 2.14 RC servo[zdroj: http://www.teamsaber.com,2011]

Pre ovládanie servomotora je výhodné použit pulzne šírkovú moduláciu PWM. Podľa dĺžky vysielaného pulzu záleží otočenie servo mechanizmu vpravo, vlavo, na stred.

V softvérovej realizácii bola využitá knižnica <servo.h>, z ktorej boli použité funkcie. Na test bol vytvorený jednoduchý program, v ktorom sa servo mechanizmus

otočil štyri krát o 60°a zistil sa čas, v ktorom to dokáže spraviť.

Nasledujúci zjednodušený kód pre ovládanie:

```
#include <Servo.h>
int pohyb_servo;                  // vytvorenie premenej pre hodnotu uhlu otočenia
Servo TestServo;                  // vytvorenie štruktúry serva
void setup()
{TestServo.attach(2);}            // priradenie výstupu 2 pre servomechanizmus.
void loop()
{ for(int x = 0; x<4;x++ )                // 4 cykly otáčok
      { for(pohyb_servo = 120; pohyb_servo > 60;pohyb_servo -= 1)   //zmena uhlu
      {TestServo.write(pohyb_servo);        //  zápis hodnoty pre servo
      delay(3);}}                         // časové opozdenie
}
```

Program bol pri testovaní doplnený o sériovú komunikáciu pre výpis a test hodnôt.

Obrázok 2.15 Výpis z konzoli

2.3.3. Hallova sonda

Senzorická časť z dôvodu zmeny konštrukcia bola zmenená z inkrementálnych snímačov na Hallove senzory.

Jedná sa o polovodičovú súčiastku, ktorá reaguje na zmenu magnetického toku. Sonda má výstupné napätie v TTL logike, ktoré je vhodné pre digitálne vstupy. Zaznamenávanie je možné pomocou dvoch metód:

- V programe sa kontroluje logický stav signálu, pri jeho jednotkovej hodnote sa podmienka splní a zaznamená.

Vzorový program:

```
int pushButton = 8; // priradenie pinu
int x=0; // premenná pre počet pulzov
void setup() {
Serial.begin(57600);
pinMode(pushButton, INPUT_PULLUP); }// definícia pinu
void loop() {
 if(digitalRead(pushButton)){ // kontrola logickej jednotky
while(digitalRead(pushButton) == HIGH) {delay(2);} //čakanie kým je v log.1
x++;   delay(2);
 Serial.println(x); delay(2);}}
```

Ďalšou variantou využívanou aj pri riešení algoritmu ABS je možnosť využitia hardvérového prerušenia. Jedná sa o externé hardvérové prerušenie, v ktorom je možné zadefinovať, na ktorý podnet má nastať. Tri podnety sú nástupná, vzostupná hrana alebo zmena hodnoty. Najlepším výberom je zmena hodnoty, pri ktorej sa pri jednej otáčke dostane dvojnásobný počet pulzov:

Vzorový program:

```
unsigned int x=0; // vytvorenie premenej pre počet pulzov
void setup()   {
attachInterrupt(0, blink, FALLING); // nastavenie prerušenie
Serial.begin(57600); // nastavenie sériovej komunikácie }
void loop()   {
Serial.println(x);} // výpis do konzoly
void blink()   { // zavolaná funkcia od prerušenia
x++;}
```

2.3.4. Podporná doska

Návrhom dosky je vyriešených v elektronickej časti modelu niekoľko problémov s napájaním:

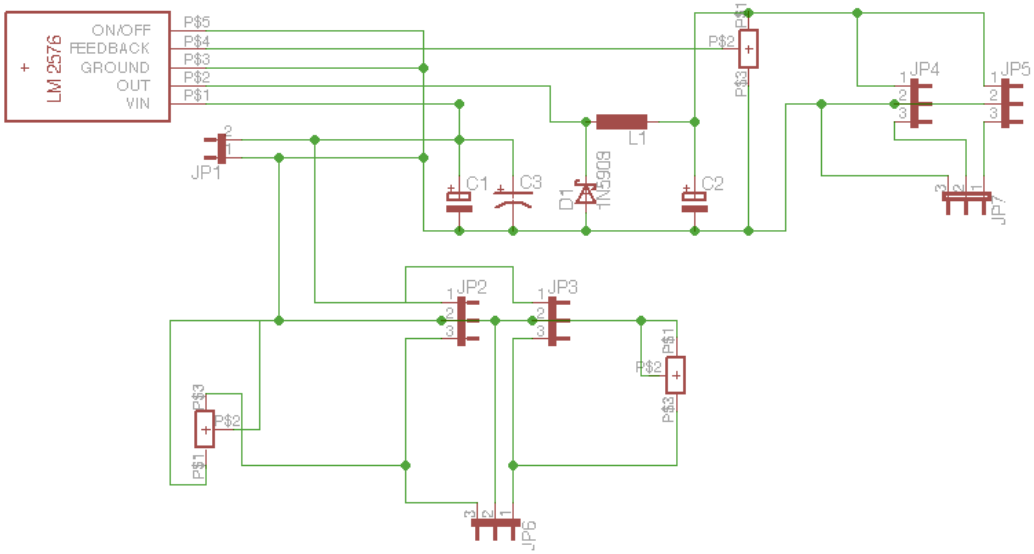
- servomechanizmi sú napájané 6V
- každá Hallova sonda má reguláciu výstupného napätia
- jednotné napájacie napätie pre dosku 12V.

Napájaciu časť pre servomechanizmi tvorí spínaný zdroj. Oproti lineárnemu zdroju má niekoľko nasledovných výhod uvedených v tabuľke 2.7 [Praktická realizace symetrického stabilizovaného zdroje,2013]

Tabuľka 2-7 Porovnanie zdrojov

parameter	spínaný zdroj	lineárny zdroj
účinnosť	75%	30%
veľkosť	0,2[W/cm^3]	0,05[W/cm^3]
cena	konštantná	rastie s výkonom

Použitým zdrojom je integrovaný obvod LM 2576 ADJ. Obvod pracuje podľa schémy uvedenej v datasheete výrobcu. Regulovateľné výstupné napätie určené servomechanizmom je na pinoch JP4, JP5 a k ním pripadá pin JP7, kde sú pripojené digitálne výstupy z Arduina. Dané riešenie umožňuje podľa potreby rýchlu zmenu sérv. Hallove sondy sú napájané priamo zo vstupného napätia podpornej dosky. Ich výstup určený pre Arduino je regulovaný pomocou trimrov. Zapojenie je na obrázku 2.16.



Obrázok 2.16 Schéma plošného spoja v programe EAGLE

Doska plošných spojov je uvedená v Prílohe A.

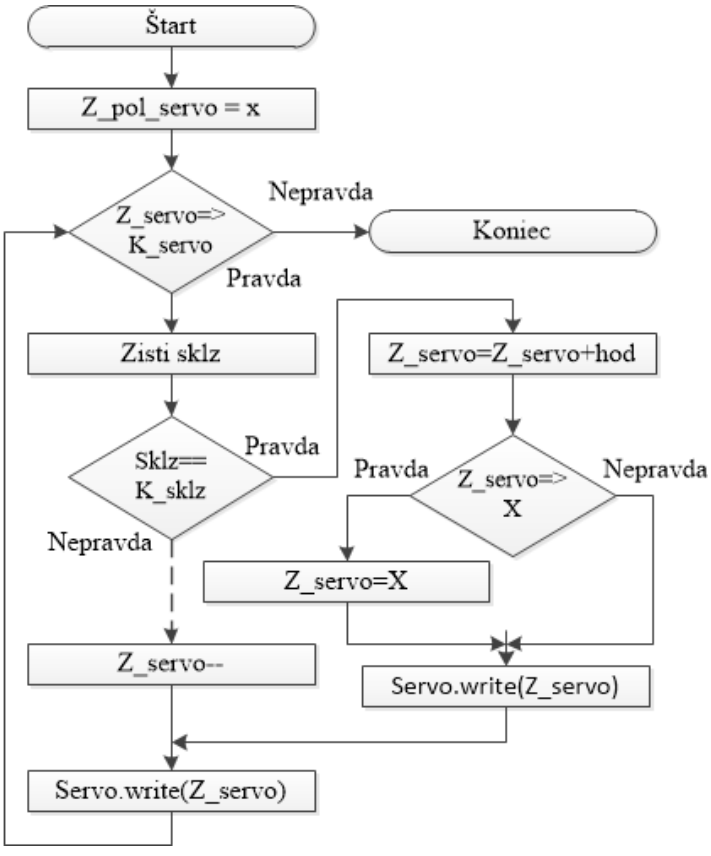
3 Vývojový algoritmus, riadiaci a ovládací softvér

Pri vývoji akéhokoľvek softvéru, či už ovládacieho alebo riadiaceho, je vhodné začať od vývojového diagramu. Pomocou diagramu si môžeme graficky znázorniť jednotlivé cykly a podcykly programu, a tým lepšie optimalizovať zdrojový program.

Vytvorené algoritmy pre model vychádzajú z algoritmu, ktorý ostal len v teoretickej rovine. Dôležitým rozdielom medzi teoretickým algoritmom a navrhnutými algoritmami je zmena umiestnenia výpočtu sklzu. Výpočet sklzu pri teoretickom algoritme je uvažovaný až po zabrzdení kolesa. Touto variantou výpočtu je jasná možnosť vytvorenia sklzu. Z logického hladiska nemusíme sklz ani počítať a brzdné ústrojenstvo pustiť.

3.1. Algoritmus ABS

Zmenenou ideou pre teoretický návrh vznikol algoritmus na obrázku 3.1. Spustenie algoritmu nastaví počiatočnú hodnotu serva Z_servo, ktorá je definovaná pre nezabrzdenú brzdu. Premenú začne inkrementovať a posielať ju servu ovládajucemu brzdu až po hranicu K_servo limitnú pre zabrzdené čeluste.



Obrázok 3.1 Algoritmus ABS

Každou prebehnutou inkrementáciou sa skontroluje sklz funkciou Zisti sklz. Vyhodnotenie výpočtu sklzu prejde v určenom reťazci podmienok.

Pre každú podmienku platí nová hranica polohy serva. Tá je kontrolovaná, aby sa poloha neposunula až za počiatočnú hranicu Z_Servo.

Softvérové riešenie algoritmu vychádza z ovládania servomechanizmu

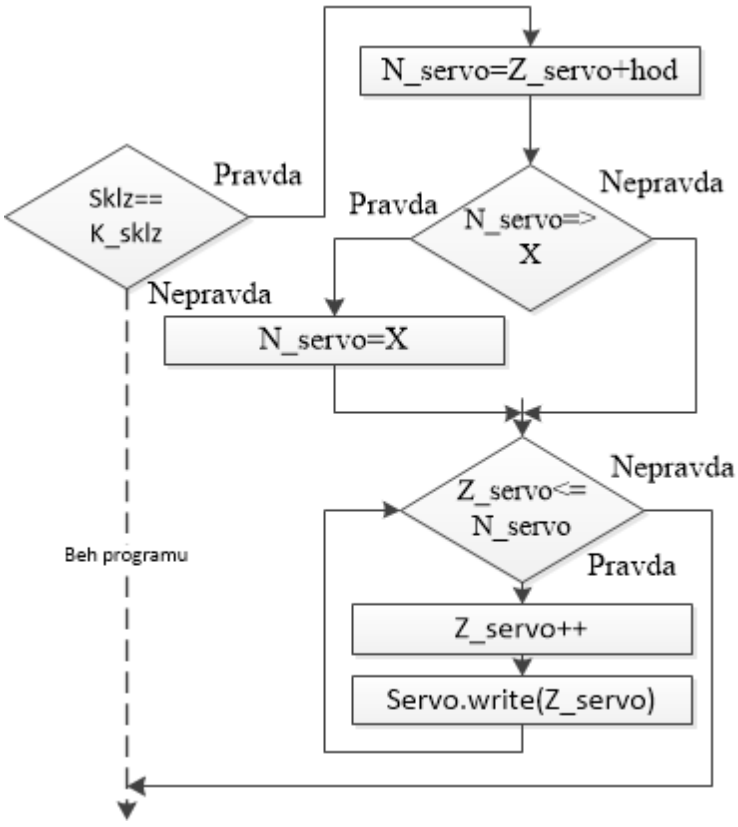
pomocou slučky. Táto slučka po vykonaní výpočtov kontroluje buď pomocou if-else, alebo príkazu switch, v ktorej hodnote sklzu sa koleso nachádza.

Názorná ukážka programu z arduina pre jednu podmienku sklzu:

```
switch(sklz){
  case 1:                    // podmienka ak je rozdiel pulzov 1
   pohyb_servo+= 38;         // návrat serva o hodnotu 38
   if (pohyb_servo>85)       // kontrola počiatočnej hranice serva
    { pohyb_servo = 85; }
 break;      // ukončenie podmienky
```

3.2. Modifikovaný algoritmus ABS

Zmenami algoritmu popísaného v kapitole 3.1 prešla len časť programu nachádzajúca sa v každej podmienke kontrolujúcej sklz. Po kontrole sklzu sa nastaví nová poloha pre servo. Prejde kontrolou, aby nebola posunutú až za počiatočnú polohu.



Obrázok 3.2 Modifikovaný algoritmus

Začne sa nový cyklus vykonávajúci sa od polohy, v ktorej bol zistený sklz až po novú polohu zadefinovanú v podmienke.

Použitím danej varianty nastalo zrýchlenie posuvu pre servomechamiznus.

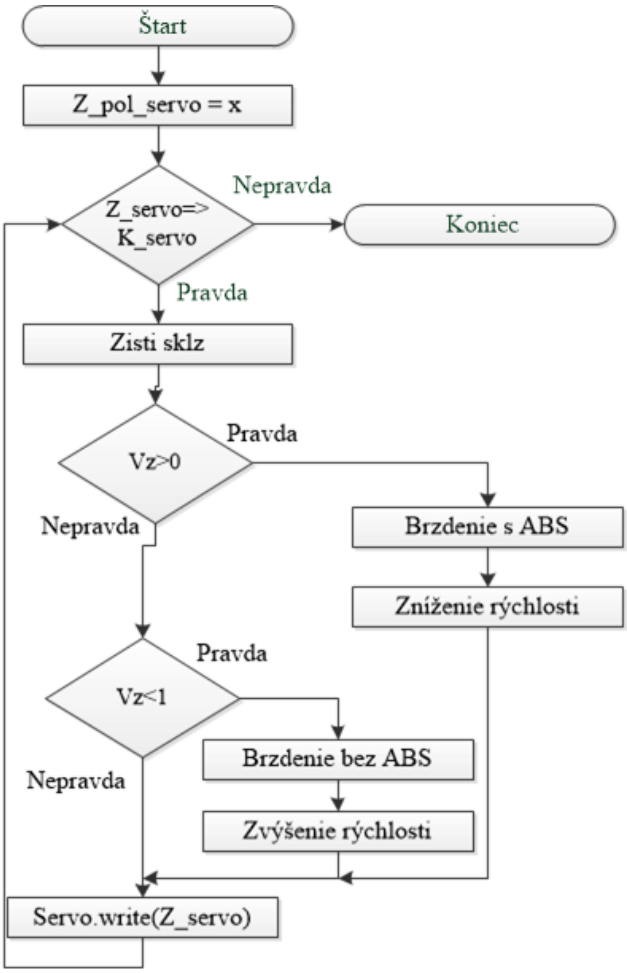
Názorná ukážka programu z arduina pre jednu podmienku sklzu:

```
switch(sklz){
 case 1:
    new_servo = pohyb_servo + 20;
    if (new_servo>85)
```

```
{ new_servo = 85;}
for (;pohyb_servo<new_servo;pohyb_servo++)
{TestServo.write(pohyb_servo);
 delay(2);
}
break;      }
```

3.3. Modifikovaný algoritmus s možnosťou náhodného zastavania

Pridáva do programu ďalšiu časť, ktorou je riešená problematika brzdenia pri nízkych rýchlostiach.



Obrázok 3.3 Algoritmus s náhodným dobrzdením

Telo hlavného cyklu má naviac dve podmienky s rýchlosťou zotrvačníku vypočítanou vo funkcii Zisti sklz. Pri rýchlosti väčšej ako nula je brzdené s ABS algoritmom uvedenom a popísanom v kapitole 3.2 doplneným o zníženie rýchlosti pomocou zmeny sklzu medzi hnacím kolesom a zotrvačníkom pomocou druhého serva.

Zvýšeným sklzom poklesne referenčná rýchlosť, čím sa pustí druhá podmienka, kde sa zatiahne brzda a koleso brzdí šmykom.

Po vykonaní udalosti sa vráti sklz medzi hnacím kolesom a zotrvačníkom späť a cyklus pokračuje brzdením s ABS. Definovaná možnosť, pri ktorej nastane zníženie rýchlosti, je definovaná pri tvorení algoritmu za konštantný počet krokov. Po odladení algoritmu je použitá náhodná možnosť.

Funkcia pre brzdenie bez ABS je obdobná ako pri definícii ovládania servomechanizmu.

3.4. Snímanie dát a priemerovanie

Pre snímanie dát je vytvorená funkcia, pracujúca s dátami z hardvérového prerušenia. Zavolaním funkcie sa uloží prvá hodnota a program príkazom *delay();* a hodnotou v ňom zadanom zastane. Externé hardvérové prerušenie beží ďalej, čím sa mení hodnota pulzov. Po skončení pozastavenia sa spraví rozdiel medzi hodnotami. Výsledkom je počet pulzov za snímaný čas. Pri kontrole dát boli viditeľné rozdiely medzi časovými intervalmi. Tieto rozdiely je si možné vysvetliť z dôvodu ručne lepených magnetov, možným preklzom medzi hnacím kolesom a zotrvačníkom.

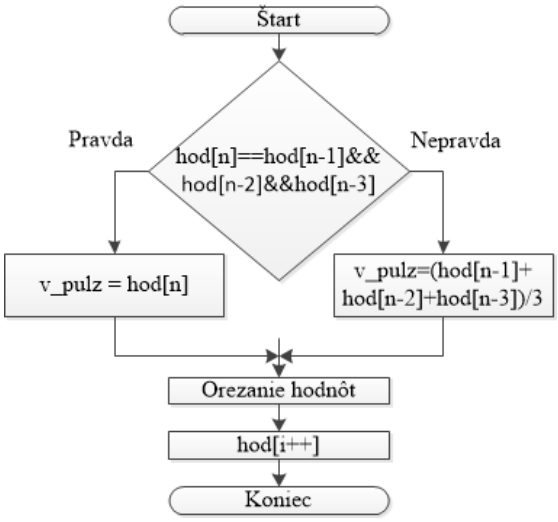
Obrázok 3.4 Snímanie dát

Z dôvodu nekonštantných hodnôt bola do programu zavedená funkcia priemerujúca hodnoty z intervalov.

Ukážka kódu pre snímanie dát v Arduinu:

```
void imp_test()
        { pulz_p = x;
        delay(25);
        pulz_c = x;
        pulz_f = pulz_c - pulz_p;
        average_puls_zotrv();
        sklz = ((int)((zpokus*3.25)+0.80))-((int)(pokus)); }
```

Funkcia priemerovania vychádza z navrhnutého algoritmu na obrázku 3.4. Algoritmus kontroluje na začiatku svojho procesu určitý počet naposledy prijatých hodnôt. Počet prijatých dát sa líši podľa nasadenia a testovania na modeli.



Obrázok 3.5 Algoritmus úpravy dát senzorov

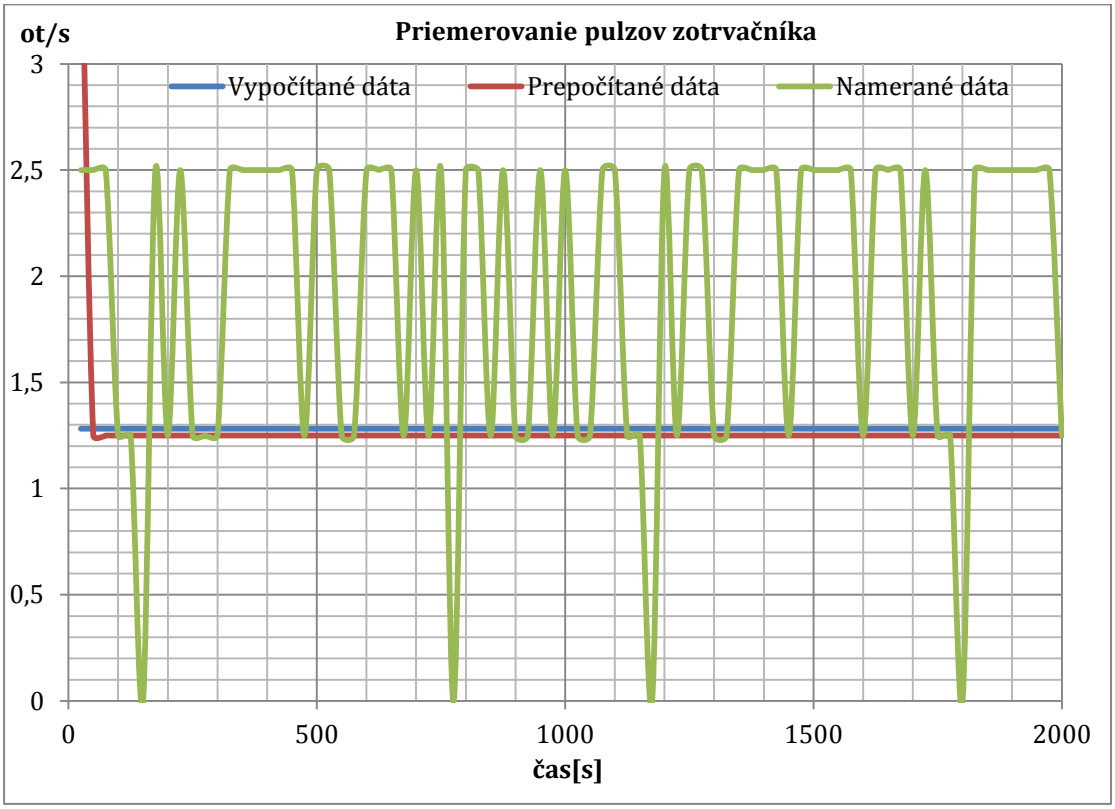
Priemer hodnôt sa vykoná za podmienky, ak sa daný počet naposledy prijatých hodnôt nezhoduje. Pri nevykonaní priemerovanie nastane, že sa na výstup priradí prijatá hodnota.

Nadpriemernej hodnote sa priradí priemerná hodnota.

Následná ukážka z Arduina:

```
void average_puls_zotrv(){
if(zkoleso_1&&zkoleso_2&& zkoleso_3==pulz_f)            // hodnoty sa rovnajú
  {       zpokus = pulz_f;       }
else if(zkoleso_1&&zkoleso_2&&zkoleso_3!=pulz_f)        // hodnoty sa nerovnajú
  {       zpokus = (zkoleso_2+zkoleso_1+zkoleso_3+pulz_f)/4;       }
if (zpokus==2){                                         // orezanie nadpriemerných čísel
    zpokus = 1;}
 zkoleso_1 = zkoleso_2;                                 // posun dát
 zkoleso_2 = zkoleso_3;
 zkoleso_3 = pulz_f;
}
```

Rotácia dát medzi premenými je v jednoprocesorovej doske z dôvodu malého množstva vnútornej pamäte a nemožnosťou vytvoriť dynamický zápis do jednorozmerného poľa tvorená pomocou rotácie medzi n - požadovaným množstvom hodnôt. Pre uvedený časový interval na obrázku 3.6 je použité priemerovanie pre zotrvačník.



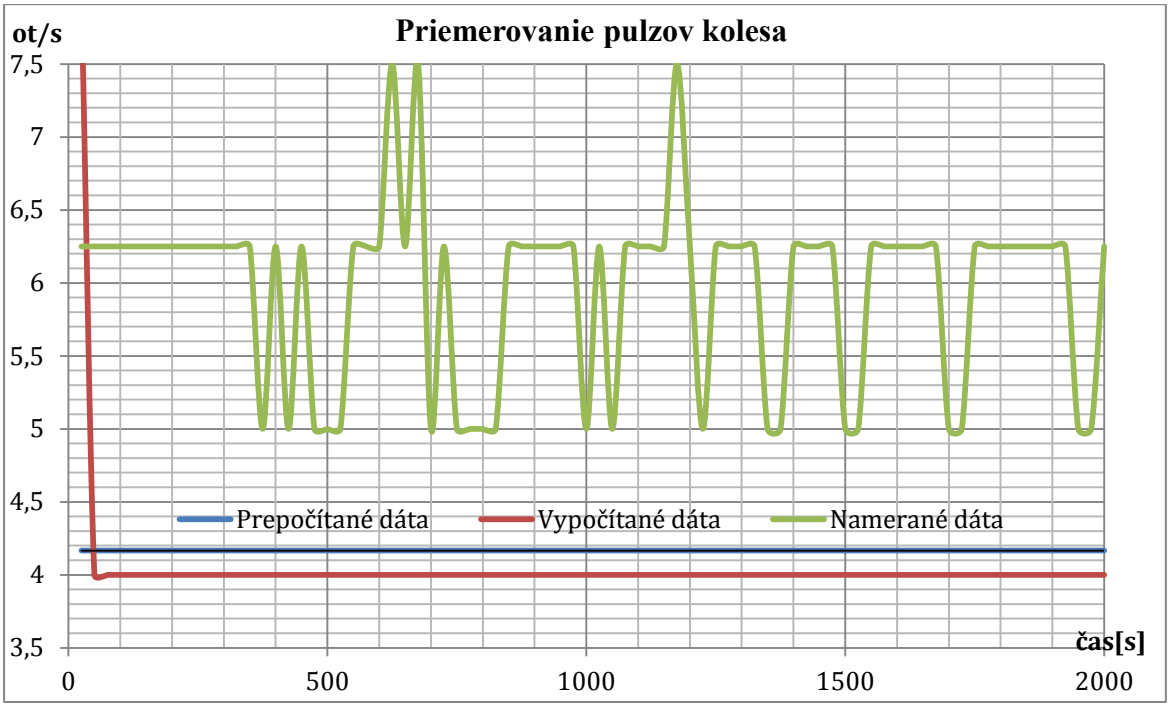
Obrázok 3.6 Priemerovanie pulzov zotrvačníka

Prijaté resp. namerané dáta sú prijaté hodnoty za uvedenú časovú slučku, dáta

nie sú vhodné pre riadenie modelu. Dáta označené ako prepočítané dáta sú výsledkom priemerovacej funkcie za Arduina. Vypočítané dáta sa týkajú za predpodkladu konštantných otáčok a prepočítania, že medzi zotrvačníkom a motorom nenastane žiadny sklz.

$$ot_z = prijaté_{pulzy} \frac{počet\ pulzov\ na\ magnetov}{časový\ úsek} = 1.\frac{32}{25} = 1{,}28\ ot/s \qquad (7)$$

Obdobný spôsob výpočtu môžeme použiť aj pre brzdiace koleso. Za takého istého predpokladu , že medzi žiadnym prevodom nenastane sklz a nepoklesnú otáčky motora.



Obrázok 3.7 Priemerovanie pulzov zotrvačníka

Použité značenie pre graf pulzov kolesa vychádza z obrázku 3.6 Prepočítanú hodnotu vyjadríme :

$$ot_k = prijaté_{pulzy} \frac{počet\ pulzov\ na\ magnetov}{časový\ úsek} = 3.\frac{32}{25} = 4{,}28\ ot/s \qquad (8)$$

3.5. Ovládací softvér

Na vyhodnotenie dát nameraných jednoprocesorovou doskou ich bolo treba dostať z jednoprocesorovej dosky do počítača. Použitá doska neobsahuje žiadny slot pre externú flash pamäť a po sériovej linke nedokáže vytvoriť súbor a zapisovať do neho.

Riešením totho problému sa zrodila aplikácia, ktorá by tento vzniknutý problém riešila.

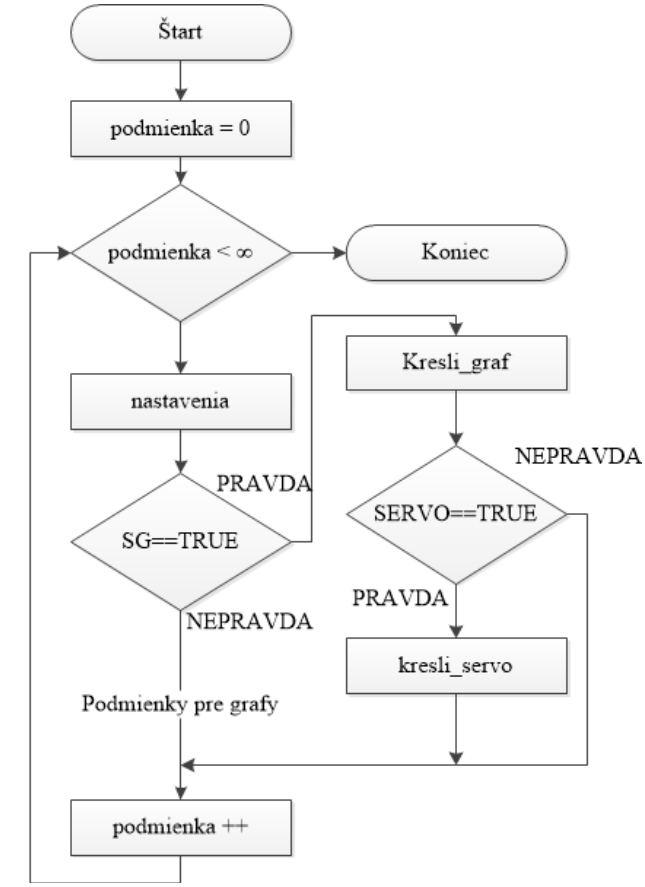
Softvér je vyvinutý v programovacom prostredí Processing IDE. Prostredie má rovnaké rozhranie ako Arduino IDE.

Výhodami Processingu sú :

- 2D, 3D a openGL grafické komponenty
- multiplatformové prostredie
- široké spektrum ponúkaných knižníc

3.5.1. Vývojový diagram vytvorenej aplikácie

Kódy aplikácií napísaných v processingu môžeme rozdeliť na dve časti. Jednou z častí je slučka *void setup() {telo cyklu},* ktorá sa vykoná len pri inicializácií aplikácie a v ďalšom behu programu dané premené nie je možné meniť.



Obrázok 3.8 Beh aplikácie

Podstatnejšou slučkou v napísaných aplikáciách je slučka void draw() {telo cyklu} vykonávajúca v nekonečnom cykle.

Algoritmus pre vyvinutú aplikáciu beží vo spomínanej slučke nasledovne. Na začiatku behu programu sa inicializujú nastavenia, v ktorých je prvotné definovanie farieb, v ktorých sa aplikácia pustí. V nekonečnej slučke sa čaká na zmenu niektorej z logickej hodnoty pre vykreslovanie grafov. Možnými výbermi sú:

- možnosť vykreslujúca grafy pulzov pre zotrvačník a hnacieho kolesa
- možnosť vykresslujúca grafy rýchlosti zotrvačníka a hnacieho kolesa
- možnosť vykreslujúca graf rýchlosti zotrvačníka a hnacieho kolesa s grafom sklzu medzi ich rýchlosťami.

Doplnkovou možnosťou pre každý výber je vykreslenie polohy serva ovládajúceho brzdu hnacieho kolesa. Pre vykreslovanie dát je nutné z jednoprocesorovej dosky získavať dáta. Nutné dáta pre beh aplikácie sú z Hallových sond, vypočítaným sklzom a polohe servo mechanizmov. Odosielanie rozličných dát v jeden okamžik je zabezpečené súborom dát.

Použitý kód v Processingu:

```
String message = myPort.readStringUntil(LF);     //načítanie reťazca po nový riadok
    if(message != null)                          // pokiaľ nie je správa prázdna
    {  String [] data  = message.split(",");     // spojenie dát
      if(data[0].charAt(0) == HEADER)   // kontrola hlavičkového súboru
      { for( int i = 1; i < data.length-1; i++)
        {numbers[i] = Integer.parseInt(data[i]);} }} // vytvorenie pola dát
```

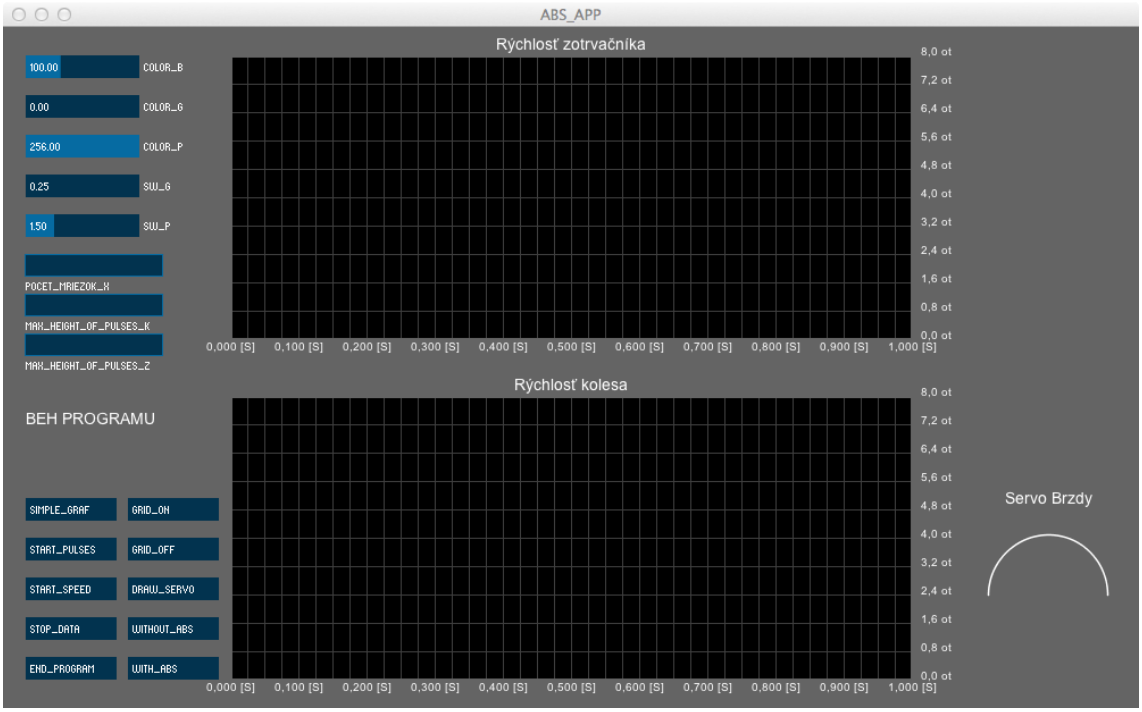
Zobrazená komunikácia bola určená pre odosielanie troch čísel. Úpravou kódu je možné posielať lubovoľný počet čísel. [Arduino CookBook,2013]

Ovládanie možnej simulácie, ktorá ma byť na modeli realizovaná, je daná zápisom na seriovú linku pomocou príkazu.

myPort.write(číslo);

3.5.2. Grafické prostredie

Posledná modifikácia z vytvorených aplikácii je na obrázku 3.9.



Obrázok 3.9 Grafické prostredie aplikácie

Vývojom každej aplikácie vzniká niekoľko výsledných verzií. Pri danej aplikácii boli verzie len dve a niekoľko ich modifikácií. Aplikácia, v prvej fázy vývoja používala ako ovládacie tlačidlá vykreslené pomocou obdĺžnikov a textov v nich napísaných. Problémom používania tejto metódy bolo pri stlačení tlačidla kontrolovať polohu myši pre každý prvok zvlášť. Po tejto kontrole sa vykonala príslušná funkcia.

Efektívnejším spôsobom je použiť knižnicu, v ktorej sa potrebné prvky nachádzajú. V programe je importovaná open-source knižnica controlp5, v ktorej sa nachádza množstvo ovládacích prvkov.

V aplikácii sú z tejto knižnice použité tlačidlá, textové polia, posuvné nastavovače hodnoty. Ovládacie komponenty vykonávajú definované funkcie potrebné pre ovládanie alebo zmenu prvotne nastavených hodnôt v aplikácii.

Obrázok 3.10 Detail tlačidiel

Tlačidlá sú využívané pre zmenou logických stavov a tým spúštanie alebo zastavovanie podčastí programov. Najčastejším využitím je ovládanie grafov. Definovanie komponent je v dvoch krokoch. Jednorázové definovanie v slučke je podmienené príkazom:

controlP5.addButton("Názov_prvku",hodnota,x,y,x_rozmer,y_rozmer);

Príkazom sa definuje tlačidlo, kde jeho názov sa napíše medzi úvodzovky, hodnota pri tutoriáloch je 1, poloha sa definuje ako súradnice x,y a následne ich rozmer v ose x a y pomocou premených x_rozmer, y_rozmer.

Pre úplné pridanie komponenty tlačidla do programu je nutné k príkazu doplniť funkciu, ktorú má komponenta v akcii vykonať, jej názov musí byť totožný s názvom tlačidla.

```
void Názov_prvku (){
        enable_p = true;        // povolenie popiskov
        start_p = true          // spustenie grafu
        myPort.write(2);}       // pustenie programu v arduinu
```

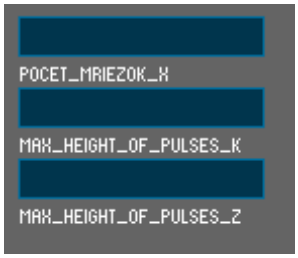
Uvedená časť funkcie je pre jedno tlačidlo, ktoré spúšťa vykreslovanie grafy pulzov. Obdobný spôsobom sú zadefinované ostatné tlačidlá.

Textové pole je prvok, v ktorom je možné zadávať znaky. V knižnici je vytvorená komponenta, ktorou je možné do programu načítať znaky, resp. textové

reťazce. Aplikáciou je využívaná hlavne pre načítanie hodnôt, podľa ktorých sú menené hodnoty pri osiach grafov. Podobne ako pri tlačidle je definovanie v jednorázovej slučke pomocou príkazu:

controlP5.addTextfield("Pocet_mriezok_x",20,200,120,20);

Pre úplné zavedenie komponenty textového poľa do programu je použitá funkcia, v ktorej sa vykoná práca s komponentou načítaným reťazcom . Z textového reťazca je možné získať len desatinné alebo celočíselné čísla. Aplikácia vo svojej činnosti využíva len celočísla.



Obrázok 3.11 Textové polia

Definovanie funkcie, ktorá je v programe využitá pre načítanie hodnôt, môžeme napísať ako:

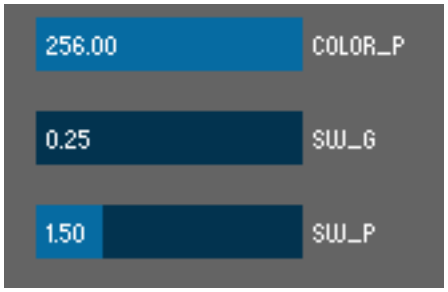
```
void Pocet_mriezok_x(String theText) { // xova os grafov
  s.graf_collor_refresh();
  if(Integer.parseInt(theText)==500){
  time = 20;}
}
```

Použité funkcie v tele ukrývajú prekreslovanie z dôvodu, aby pri zmene nastavených parametrov začali grafy kresliť od počiatočnej hodnoty. Dôvodom je prekreslenie už existujúcich grafov a lepšej prehľadnosti v grafoch. Telo funkcie je obdobné pre xsovú alebo ypsilonovú os.

Posuvný nastavovač hodnoty v informatike označený ako slider je využívaný k spojitej zmene hodnôt ovplyvňujúcich použité farby grafoch, pulzoch, pozadiach a hrúbky čiar mriežky a znázornených údajov.

Do programu je definovaný nasledovne:

controlP5.addSlider("SW_P").setRange(1,3).setValue(1.5).setPosition(20,165)
.setSize(100,20);



Obrázok 3.12 Detail posuvného nastavovača hodnoty

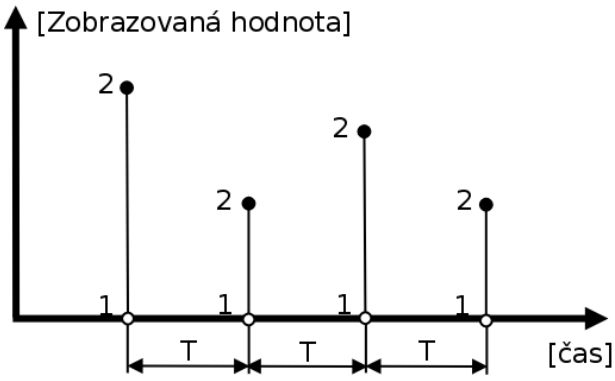
Pri definovaní tejto komponenty je možné zvoliť počiatočné podmienky a rozsah, v ktorom sa slider pohybuje. Definovanie parametrov je možné pomocou príkazov *.setRange(minimálna hodnota, maximálna hodnota)*, v tomto rozsahu je možné hodnoty meniť, jeho prvotnú hodnotu je možné ovplyvniť parametrom *.setValue(hodnota).* Hodnota uvedená v zátvorke sa nastaví pri prvom spustení aplikácie a zostane nastavená, kým nenastane jej zmena.

```
if(theEvent.controller().name()=="SW_P") {
sw_p = theEvent.controller().value();
 s.graf_collor_refresh();
}
```

3.5.3. Vizualizovanie dát

Najlepšie pre názornú ukážku činnosti a testu algoritmov je vykreslovanie získaných hodnôt do grafov. V aplikácii nie sú použité žiadne komponenty z knižníc, ale vytvorené metódy vykreslovania.

Metóda pre znázornenie hodnoty pulzov získaných v časovej slučke je tvorená ako diskrétny graf. Na xsovej osi je zobrazovaný čas a na ypsilonovej osi je zobrazovaná hodnota dát.



Obrázok 3.13 Graf pre diskrétne hodnoty (1 - xová súradnica čiary, 2 - ypsilonová súradnica čiary)

Z prvej prijatej hodnoty napr. 6 pulzov za periódu sa jednorazovo aktivuje funkcia nastavujúca rozsah, ktorá nastaví maximálnu hodnotu pre os 12.

Po jej zadaní rozsahu grafu vypočíta polohu pulzu nasledovne.

$$\text{poloha\_y} = \text{x\_poloha\_grafu} - \frac{prijaté\ dáta}{zvolený\ rozsah}\ \text{dĺžka yps. osi} = 275 - \frac{6}{12}250 = \square\square\square\ [\mathbf{px}] \quad (9)$$

Poloha xsovej osi nie je zisťovaná, ale je previazaná z časom slučky v Arduinu. Nastavenou hodnotou je zobrazovanie 1 sekundy . Pre túto nastavenú hodnotu výpočet prebieha:

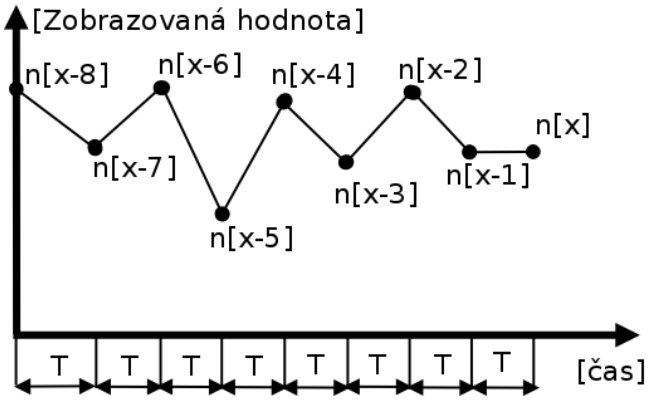
$$\text{poloha\_x} = \text{poloha\_x} + \frac{\text{dĺžka xsovej osi}}{\text{počet zobrazených údajov}} = 200 + \frac{600}{40} = \mathbf{215}\ [\mathbf{px}] \quad (10)$$

Odpočítaním polohy od ľavého okraja celej aplikácie zistíme, že pulz sa bude vykreslovať každých 15 pixelov. Výpočty uvedené v rovniciach 9 a 10 sa opakujú pre každé nové dáta. V úryvku programu je naviac definícia farieb, hrúbok čiar, kontrola koncovej polohy potrebná k prekresleniu grafu.

```
if(start_p)  //spustenie grafu
```

```
{ fill(myColorGraf); // pozadie grafu
  if(poloha==800||poloha == 795) // kontrola polohy pre zadané časy
  {  s.graf_collor_refresh(); // prekreslenie funkciu
     if(grid){ s.grid_graf(time,Npulz_v,Npulz_z);} // aktivovanie mriežky
     poloha = 200;} // počiatočná hodnota
  float poloha1 = int (numbers[1]);          // načítanie hodnoty
  stroke(myColorPulz); strokeWeight(sw_p);       // definícia pre čiar
  poloha1 =  (float)275 - (poloha1/ot_z)*250;      // výpočet polohy_y rovn. 9
  line(poloha, y_size_graf+250, poloha,poloha1); // vykreslenie čiary
  poloha = poloha+(600/time);}    // nový výpočet polohy_x rovn. 10
}
```

Spojitý graf využívaný pre rýchlosť kolesa, zotrvačníka alebo kolesa, zotrvačníka so sklzom je na obrázku 3.14. Rozmiestnenie na osiach je totožné s diskrétnym grafom.



Obrázok 3.14 Graf pre rýchlosť

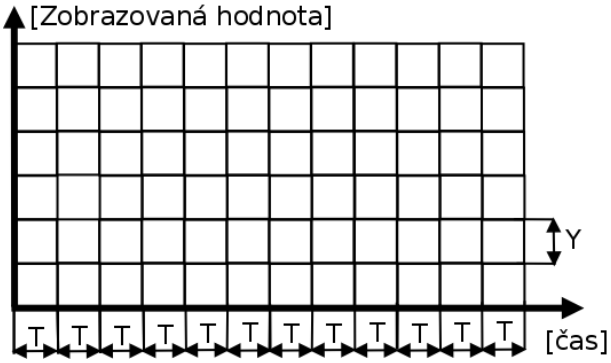
Vykreslovanie hodnôt je upravené pre tvar spojitého tvaru. Nastavanie rozsahu pre hodnoty na ypsilonovej osi je upravené kvôli možnej voľbe, kde sa v jednom grafe nachádzajú rýchlosti pre zotrvačník a brzdiace koleso. Tu v kontrole vykonáva porovnanie rýchlosti a z väčšej rýchlosti definuje rozsah pre os.

Výpočet je tvorený z aktuálnej a predchádzajúcej hodnoty a z jedného prepočtu časovej súradnice rovnakým spôsobom pixelov z predchádzajúceho riešenia. Zobrazovaná hodnota je prepočítaná z pulzov na otáčky za sekundu.

```
void graf_s(){
  if(p>3) // spustenie keď budú načítané viac ako 4 hodnoty
   { fill(myColorGraf); stroke(myColorPulz); strokeWeight(sw_p); // definície
     line(poloha, 275-((250.0/ot_z)*numbersx[p-2])*1.25,   // prvá hodnota x,y
     poloha +(600/time), 275-((250.0/ot_z)*numbersx[p-1])*1.25); // druhá hod. x,y
       poloha = poloha + (600/time);          // nový výpočet polohy
    if(poloha==800||poloha == 795)          // kontrola xsovej súradnice
    { s.graf_collor_refresh();                 // prekreslenie grafu
```

```
        if(grid){ s.grid_graf(time,Npulz_v,Npulz_z);}  // možnosť mriežky
        poloha = 200;}  // počiatočná hodnota
    }
}
```

Komplementáciou a skonfortnením pre vizualizáciu grafov je možnosť zapnutia mriežky. Tá ma pre určitý interval hodnôt nastavenú pevnú veľkosť. Jej vykreslenie prebehne po zmene logickej hodnoty a kontrole ak je nejaký grafov aktívny.

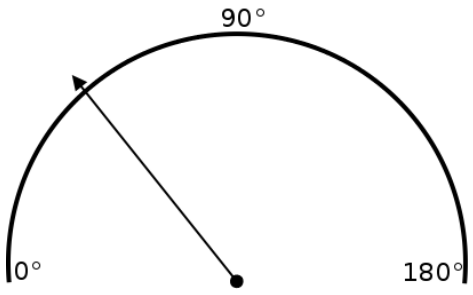


Obrázok 3.15 Vykreslovaná mriežka

Parametre pre mriežky vychádzajú zo zadaných rozsahov ypsilonových osí a zvolenej časovej základne. Zvolená aktívna mriežka pre čas jednej sekundu a prvej prijatej hodnoty 6 podľa obrázku 3.13 má šírku T 15 pixelov a šírku Y definovanú z daného rozsahu 6,25 pixelu. Dané hodnoty sa dosadia do cyklov pre vykreslenie mriežky v nasledujúcom kóde:

```
void grid_graf(float gridSizeX,float gridSizeY_k,float gridSizeY_z)
 { s.graf_collor_refresh();  float pokus = 600/gridSizeX;
  for (float i = 200; i <=800; i+=pokus)
  { for (float j = 25.00; j <275; j+=gridSizeY_z)
    { fill(myColorGraf); stroke(255); strokeWeight(sw_g);
      rect(i,j,pokus,gridSizeY_z); }
```

K jednotlivým zvoleným typom grafov možno vykreslovať polohu serva a zisťovať činnosť brzdy. Servo má rozsah činnosti 0° až 180°. Prijatá hodnota sa rozloží na xsovú a ypsilonovú súradnicu pomocou goniometrických funkcií.



Obrázok 3.16 Vykreslovanie polohy ramena servomechanizmu

Vykreslovanie polohy serva je uvedené na nasledujúcom kóde:

```
void draw_servo(){
        pokus = numbers[4];
        noFill(); stroke(myColorPulz); strokeWeight(sw_p);
        arc(920, 500, radius*2+5,radius*2+5, PI, 2*PI);
```

*strokeWeight(sw_p*2);*

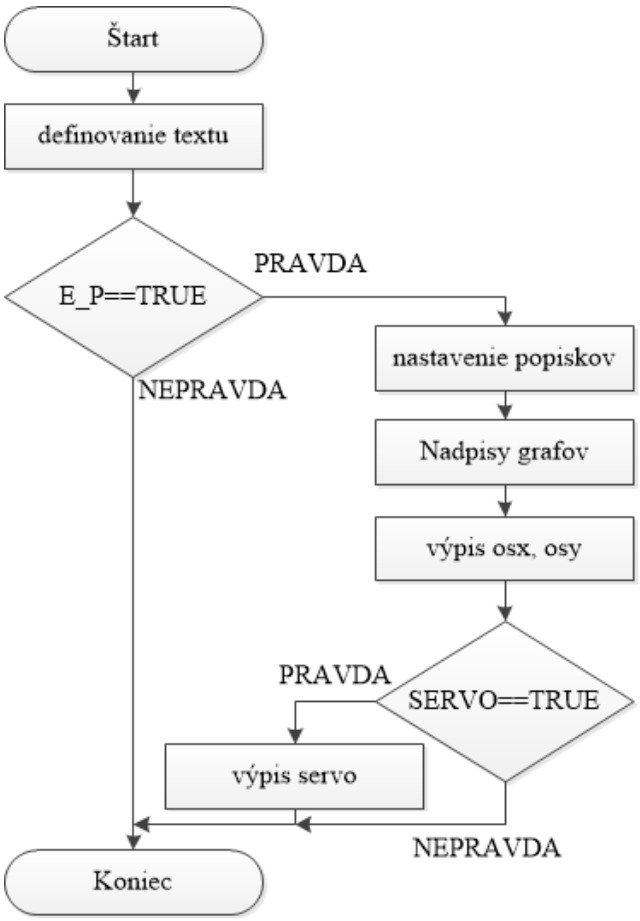
line(920, 500, 920 + cos(radians(pokus+180))(radius+3), 500 +*

sin(radians(pokus+180))(radius+3););*

}

3.5.4. Vypisovanie textu

Vo funkcii nastavanie volanej v hlavnej slučke definujúcej prvotné nastavenia pre program je ukrytý podprogram zabezpečujúci výpis textových popiskov pre uvedené grafy. Podprogram zabezpečuje úpravu rozmiestnenia popiskov k osiam. O činnosť sa starajú tri funkcie, z ktorých jedna je o ich správu s uvedeným algoritmom na obrázku 3.17 a zvyšné o výpis k osiam.



Obrázok 3.17 Algoritmus pre vypisovanie textov

Hlavnou funkciou vypisujúcou popisky je funkcia, ktorá na začiatku definuje textové parametre a keďže je vykonávaná v nekonečnom cykle čaká na zmenu logickej hodnoty.

Po zmene niektorej z logických hodnôt sa vo funkcii definujú textové parametre a ich následkom sa dvakrát zavolá funkcia vypisujúca nadpisy. Dôvod dvojnásobného volania je zabezpečenie voči nežiaducemu neprepísaniu textu.

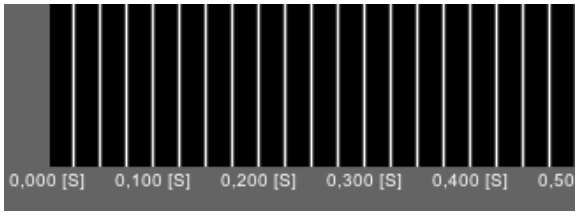
Pokračovaním v algoritme je zavolanie funkcie pre os x a os ypsilon. Servomechanizmus má definovaný len nadpis a popisky k vykresleniu polohy sa zobrazia až po zmene logickej hodnoty.

Použitý kód v Processingu:

void text_setup(logická hodnota pre pulzy , logická hodnota pre graf so sklzom , logická hodnota pre rýchlosti,int time,int arduino_time){

```
fill(myColorPulz); textSize(9); textAlign(LEFT); textFont(f,14); //definovanie textu
text(info_program,20,350); // výpis informácií behu programu.
if(enable_p){ / / spustenie vykreslovania textu pre pulzy
        title1 = "";title2 = "";fill(myColorPulz); // prázdne reťazce
        t.text_title(title1,title2); // funkcia starajúca sa o nadpisy
        title1 = "Pulzy zotrvačníka"; title2 = "Pulzy kolesa";  // reťazce pre grafy
        t.text_title(title1,title2); // funkcia starajúca sa o nadpisy
        t.text_title_osx(time,arduino_time); // nastavovanie osy x
        t.text_title_osy(osy_pulz,ot_z,ot_k); // nastavovanie osy y
        textSize(9); textAlign(CENTER); textFont(f,14);
        text("Servo Brzdy",920,420);
        if(enable_servo){
                t.servo_text();
         }
}
```

Podprogramami z funkcie definovanej ako hlavná sú programy pre výpis popiskov k osiam. Popisky k osiam sú previazané z hodnotami definícií uvedených v kapitole 3.5.3. Časová os pre obidva grafy je nastavovaná z jednej funkcie a rozsah ypsilonovej osi je možné nastaviť nezávisle pre grafy.



Obrázok 3.18 Dynamické popisky pre xsovú os

Na nastavenie popiskov osi je potrebné upraviť vypočítanú polohu súradníc dát pre grafy tak, aby boľa vhodná pre vypísanie textu. Tento problém rieši fakt, ktorý sa odzrkadloval tým, že sa text prepisoval cez seba. Budeme uvažovať zadaný čas 1000 ms, k nemu určenú konštantu time 40 z kapitoly 3.5.2 a hodnotu násobiča, ktorou je udávaný počet, po ktorom sa má popisok zobraziť. Hodnota je nastavená pre každú zvolenú mierku samostatne. Pre 1000 ms je to 4. Výpočet je definovaný nasledovne.

$$\text{rozmer1} = \frac{\text{dĺžka xsovej osi}}{\text{čas}} \quad \text{násobič} = \frac{600}{20}.2 = \mathbf{60}\ [\mathbf{px}] \tag{11}$$

$$\text{krok} = \text{arduino\_time násobič} = 25.4 = \mathbf{100}\ [\mathbf{ms}] \tag{12}$$

Výpočet môžeme overiť na obrázku 3.18, kde vidíme výrez z grafu pulzov s časom 500 ms z celkového času 1000 ms, že popisky sú uvedené pri každej štvrtej

hodnote. Hodnota výsledného výpisu k osiam je kontrolovaná a pri prekročení definovanej hodnoty zmení vypísané popisky.

Ukážka kódu pre xsovú funkciu:

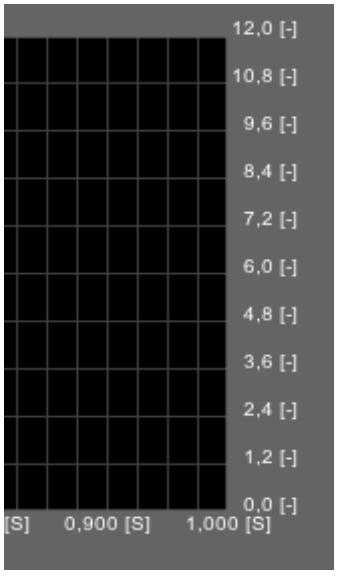
```
void text_title_osx(int cas,int arduino_time){
int rozmer, krok,nasobic=1,new_krok = 0,delic=1; float rozmer1; String osx = "",
osx1 = "[mS]",osx2 = "[S]";          // string pre úpravu popiskov
if((cas == 20)          { nasobic = 4; }
rozmer1 = (600.0 / cas)*nasobic;
krok = arduino_time*nasobic;
for (int i = 200;i<=800;i+=rozmer1){
textFont(f,10); textAlign(CENTER);
 if(arduino_time*cas<1000){
        delic = 1;
         osx = nf((new_krok/(1.0*delic)),1,0);
        text(osx+" "+osx1,i, 285); text(osx+" "+osx1,i, 585);
 }}
```

Popisky majú pre prvý výpis nastavenú hodnotu pre možnosť nespustenia simulácie alebo neprijatia žiadnych dát. Tá je prepísaná podľa vykonanej kontroly spomínanej funkcie pri nastavovaní rozsahu pre vykreslovovania osí.

Zmenou rozsahu je možné prehustenie vypísaného obsahu, a tým vzniká situácia neprehľadných popiskov.

Definovaný výpočet si predvedieme pri prvej hodnote 6 kde funkcia automatický nastaví rozsah na hodnotu 12.

Pre nastavený rozsah 12 je v programe nastavená výsledná veľkosť mriežky 6,25 a použitý násobič 5. Výpočet prebehne nasledovne :



Obrázok 3.19 Dynamické popisky pre ypsilonovú os

$$\text{rozmer} = \text{Npulz\_znásobič} = 6{,}25.4 = \mathbf{25}\ [\mathbf{px}] \quad (13)$$

$$\text{krok}_{\text{zot}} = \frac{nast\_hod\_osdefi.\_vzdial.}{dĺžka\ osi\ y}\ \text{násobič} = \frac{12.6{,}25.5}{250} = \mathbf{1},\square\ [-] \quad (14)$$

Prebehnutý výpočet sa vykonáva cyklicky a je menení na začiatku volania jednotlivých grafov, alebo pri zmene zadaného rozsahu.

Ukážkový výpočet pre načítanú hodnotu 6 nastavil vykreslovanie pre hodnotu 12 s krokom 1,2.

Na osi y pre hodnotu 6 bude 10 hodnôt s krokom 0,6.

Ukážkový kód z funkcie:

```
void text_title_osy(String ylabel,float rozmer_zot, float rozmer_kol){
 float krok_zot = 0.0,krok_kol = 0.0;  int nasobic_kol = 1, nasobic_zot = 1, rozmer;
 float new_krok_kol = 0.0,new_krok_zot = 0.0; String zot="",kol="";
 if ((250/Npulz_z)==40)        { nasobic_zot = 5;}
 rozmer  = Npulz_z*nasobic_zot;
krok_zot = (rozmer_zot / (250/Npulz_z))*nasobic_zot;
for (int i = 25;i<=275;i+=(rozmer)){
if((abs(rozmer_zot-new_krok_zot))<=10||(abs(rozmer_kol-   new_krok_kol))<=10){
        zot = nf(abs(rozmer_zot - new_krok_zot),1,1);}
textAlign(RIGHT); textFont(f,10); text(zot +" "+ylabel,835,i);
new_krok_zot += krok_zot;
}
```

4 Experimentálne overovania

Model má svojou koncepciou slúžiť pre overovanie algoritmov brzdenia a brzdenia s ABS. Na overovania sú použité dve metódy, pri jednej overujeme čas brzdenia podľa simulácie a reálneho modelu a pri druhej nasadenie algoritmov ABS s porovnaním voči klasickému brzdeniu.

4.1. Čas brzdenia podľa simulácie a reálneho modelu

Pri simulácii laboratórneho standu so zanedbaním odporov v ložiskách a trením kotúčovej brzdy, nepoznáme vyvodený brzdný moment. Ten určíme nasledovným experimentálnym spôsobom. Budeme uvažovať, že kinetická energia uchovaná v zotrvačníku sa rovná práci potrebnej pre zastavenie, a s tým môžeme počítať nasledovne:[Coefficient of Friction,2013][Brzdná dráha a jej výpočet,2013]

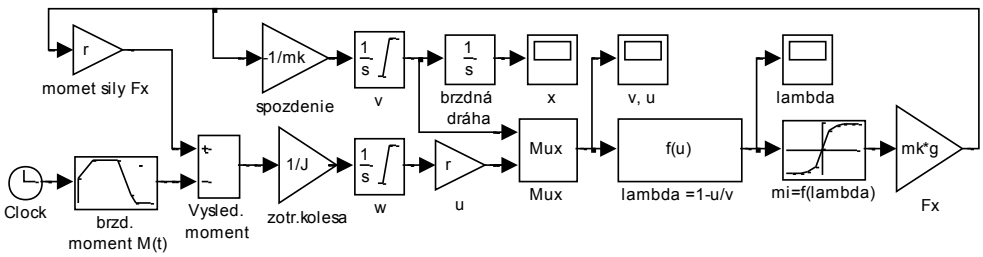
$$E_k = W_p \tag{15}$$

$$\frac{1}{2}mv^2 = \mu mgs \tag{16}$$

$$s = \frac{v^2}{2g\mu} = \frac{2^2}{2.9{,}81.0{,}6} = \square{,}\square\square\mathbf{4}\ [\boldsymbol{m}] \tag{17}$$

Vypočítanú dráhu a potrebné konštanty z kapitoly 2. dosadíme do simulačnej schémy na obrázku 4.1. [Noskievič,1999]

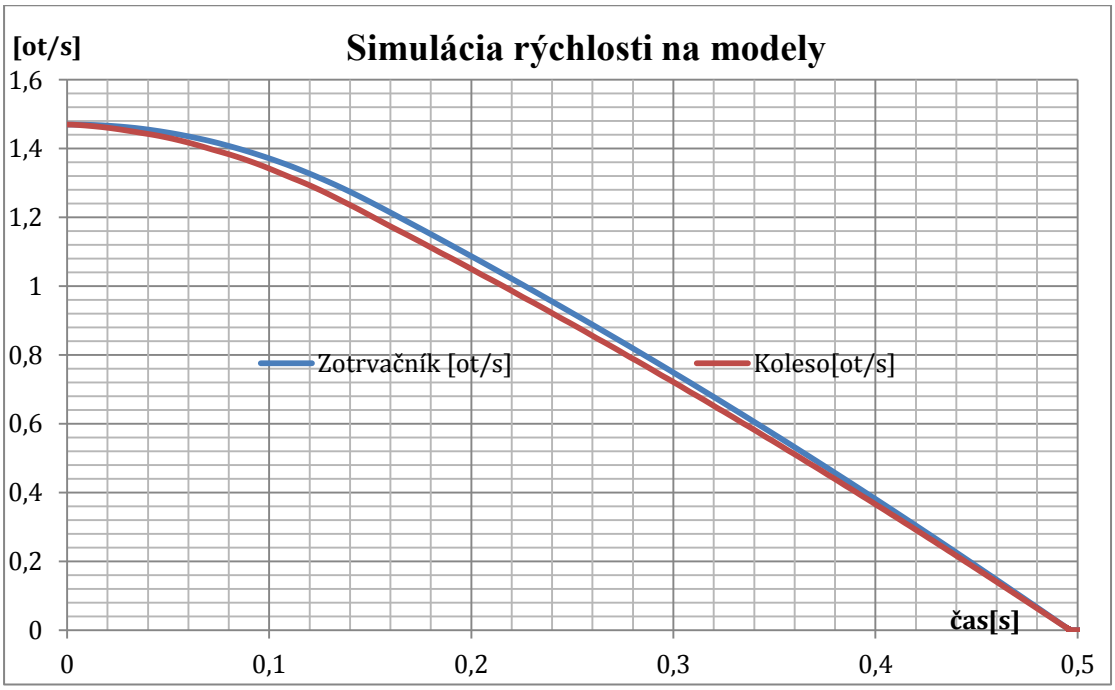
Metódou pokus - omyl nastavíme brzdný moment tak aby výsledná dráha sa zhodovala s vypočítanou.



Obrázok 4.1 Simulačná schéma modelu ABS

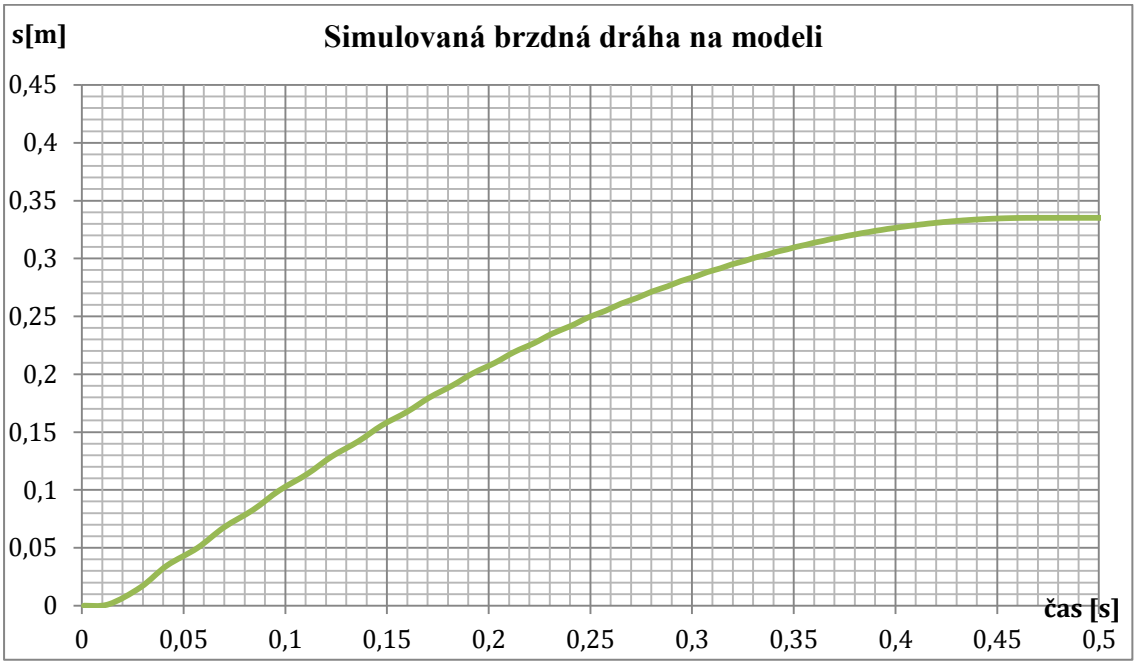
Po prebehnutí simulácii si môžeme overiť simuláciu rýchlosti na nasledujúcom obrázku 4.2, kde je vidno, že model zastane na vypočítanej dráhe s

výsledným simulačným časom 0,5 s.



Obrázok 4.2 Simulácia brzdnej dráhy a rýchlosti na modeli.

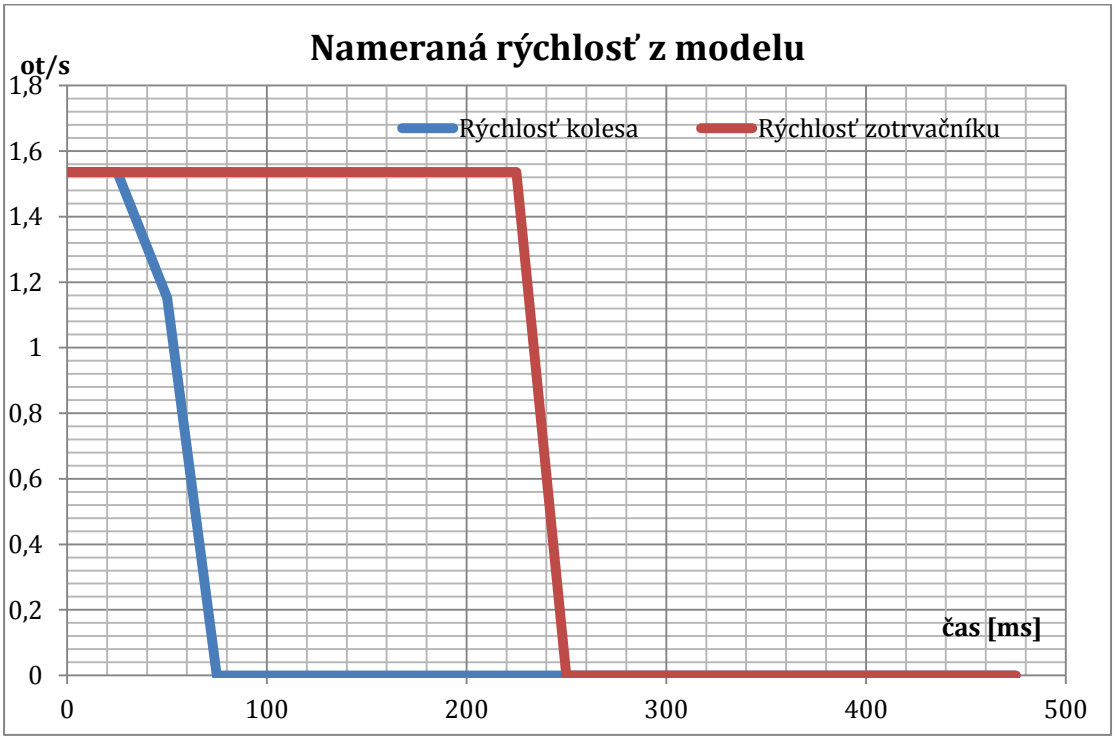
Daná simulácia je vytvorená pre vypočítanú brzdnú dráhu podľa rovníc (15)(16)(17).



Obrázok 4.3 Simulovaná brzdná dráha

Porovnanie výsledkov na modeli je tvorené podľa programu, v ktorom servo potiahne motor, tým prestane mať zotrvačník konštantné otáčky a brzdiace koleso ho

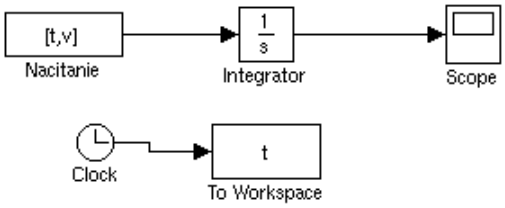
môže začať pribrzďovať. Z grafu na obrázku 4.4 je vidno, že medzi danými rýchlosťami dochádza k výraznému sklzu rýchlostí.



Obrázok 4.4 Namerané dáta z modelu

Z nameraných rýchlostí pomocou integrácie v Matlabe (obrázok 4.5) dostaneme brzdnú dráhu.

Dráha zotrvačníka sa javí ako totožná a to si môžeme vysvetliť nasledovne.

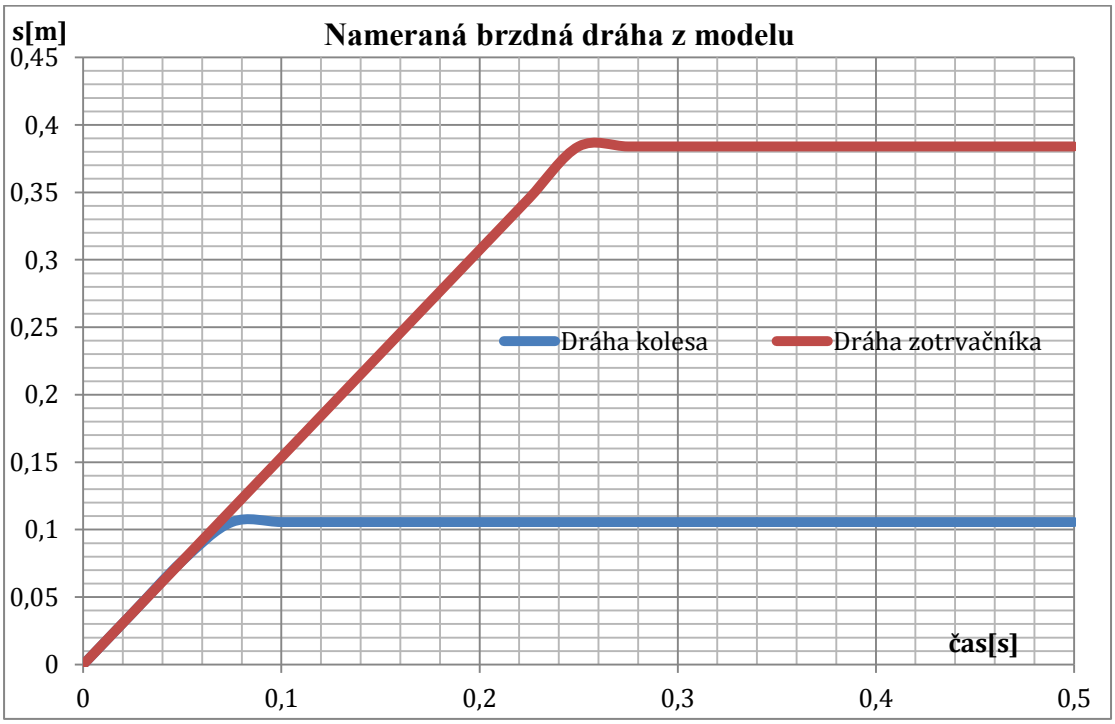


Obrázok 4.5 Prepočet rýchlosti na dráhu

1. Rozdiel v brzdných dráhach pri jednotlivých rýchlostiach je takmer totožná so simulovanou dráhou.
2. Pri simulovanom brzdení padá rýchlosť po časovom úseku 0,1s takmer lineárne a nedochádza k blokácii brzdiaceho kolesa.

Meraným priebehom na laboratórnom stande sa rýchlosti dostávajú do sklzu, čo má za dôsledok po integrácii rozdielne brzdné dráhy brzdiaceho kolesa a zotrvačníku. Rozdiely v simuláciách podľa grafov rýchlostí ukazujú na to, ako sklz nastáva a ako pri simulácii tieto rýchlosti padajú takmer s lineárnym sklonom.

Dôležitým faktorom pri reálnom brzdením je práve okamžitá blokácia brzdiaceho kolesa. Tá nastáva za 75 ms, pri celkovom čase brzdenia 500 ms je to krátky brzdný čas.



Obrázok 4.6 Brzdná dráha z nameraných dát

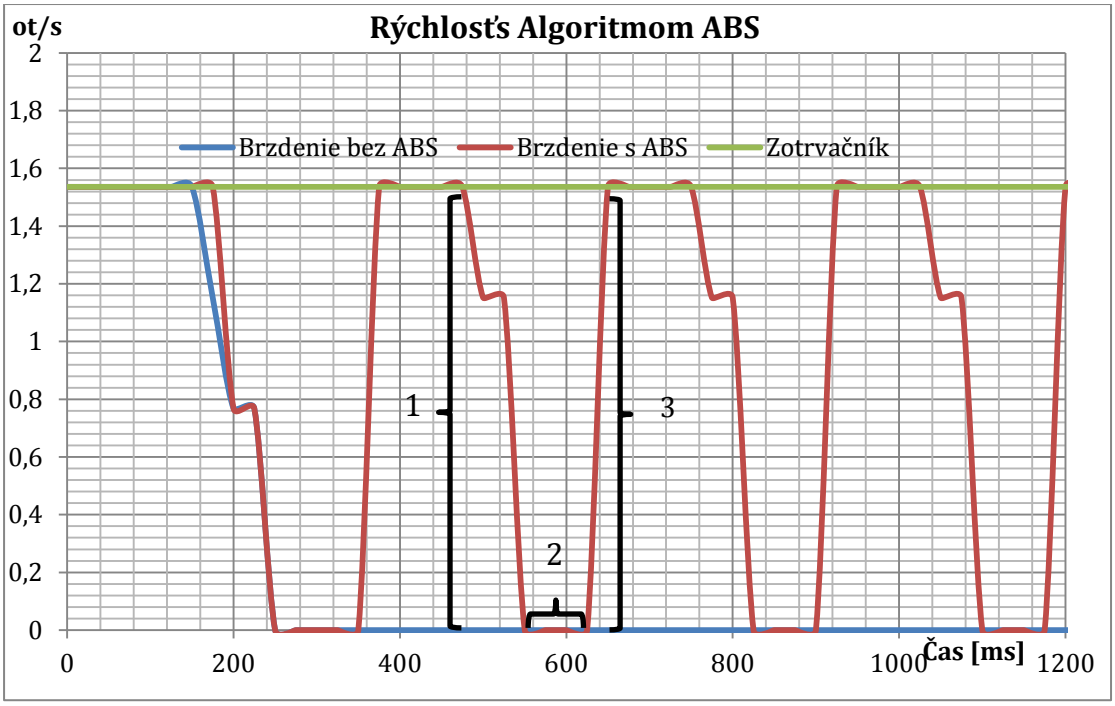
4.2. Porovnanie klasického brzdenia a pomocou ABS

Z predchádzajúcej kapitoly môžeme vidieť zo simulácie alebo reálneho merania na modeli krátky časový úsek, v ktorom model zastane.

Týmto faktom stand neslúži predovšetkým pre zastavenie zotrvačníka, ale pre kontrolované brzdenie, pri ktorom brzdiace koleso má čo najmenší sklz. Algoritmy preto budú nasadzované za konštantných podmienok pre merania. Tieto podmienky definujeme pri konštantnej rýchlosti zotrvačníka a pri nej je možné porovnávať brzdenie bez ABS alebo s ABS.

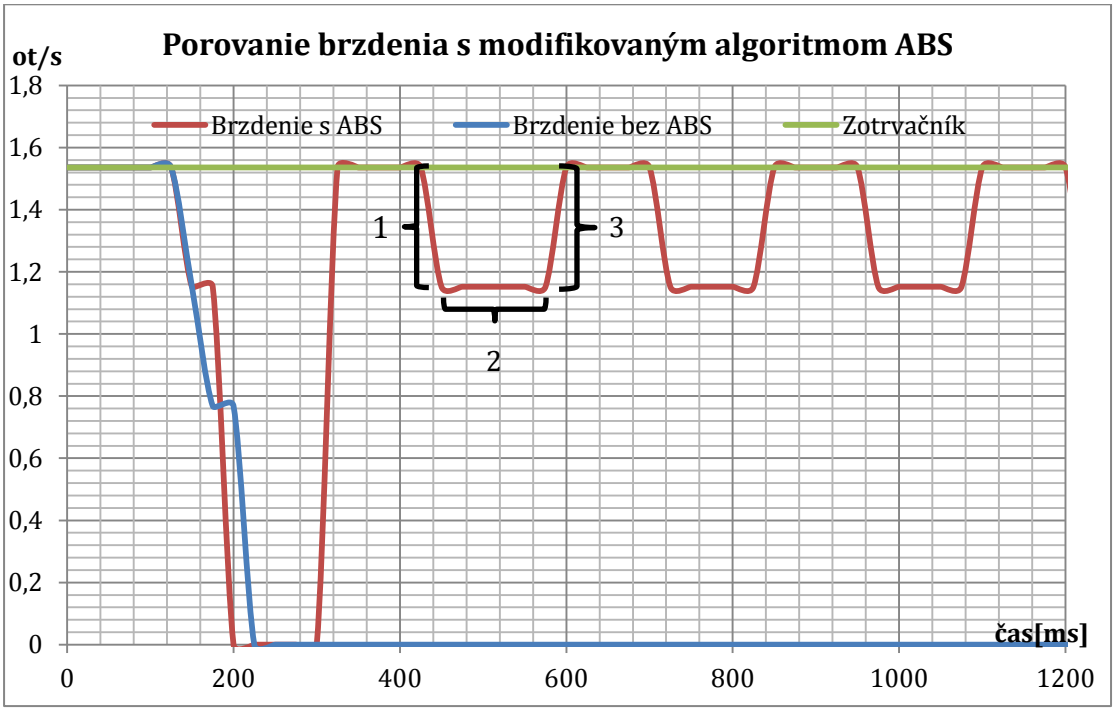
Nasadenie algoritmov môžeme vidieť na nasledovných obrázkoch 4.7 a 4.8. Ich porovnaním môžeme z grafu určiť tri časové úseky, v ktorých dane algoritmy pracujú:

- Úsek s označením 1 je interval na grafoch, pri ktorom dochádza k znižovaniu rýchlosti kolies pôsobením serva na brzdu s výsledkom vytvárania brzdnej sily v časoch 25 až 50 ms pre jednotlivý algoritmus.



Obrázok 4.7 Graf rýchlosti algoritmu ABS

- V druhom intervale sa brzdiace koleso nachádza v sklze trvajúcom od 75 do 100 ms. Rozdielom medzi algoritmy je, že nastáva maximálny sklz, ktorý pri prvom algoritme je maximálny a pri modifikovanom je priemerne 0,5 ot/s.

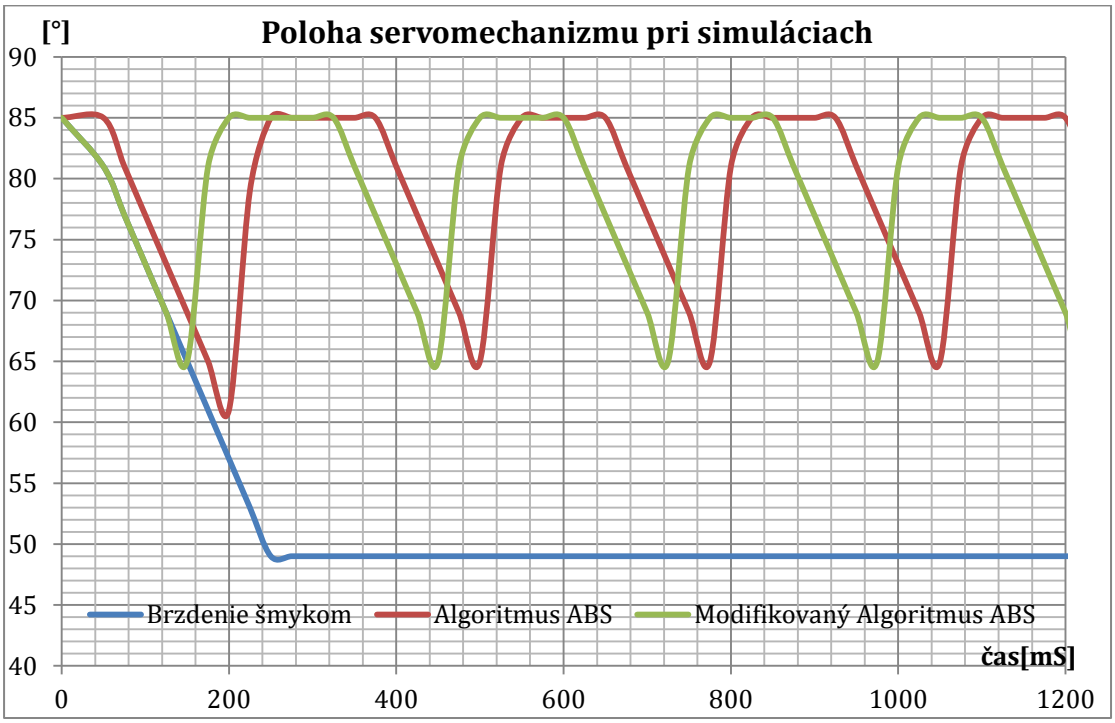


Obrázok 4.8 Graf rýchlosti modifikovaného algoritmu

- Posledný úsek s číslom 3 je časový interval, pri ktorom dochádza k zvyšovaniu rýchlosti kolies.

Modifikovaný algoritmus však dokáže brzdiace koleso držať na menšom sklze, ktorý vychádza zo zmeny ovládania serva pri zistenom sklze. Táto zmena je už spomenutým pozitívom tohto algoritmu, ktorý je rýchlejší a má kratšie reakcie pri brzdení.

Potvrdenie rýchlosti je aj pri porovnaní polôh brzdiaceho servomechanizmu. Modifikovaný algoritmus oproti jeho predchodcovi zvláda kratšie časy posunu. Z grafu je možné vidieť koncovú polohu 65°, pri ktorej je koleso v šmyku. Klasické brzdenie dotiahne servo až na jeho definovanú hodnotu.



Obrázok 4.9 Poloha ramena servomechanizmu pri brzdení

5 Záver

Cieľom diplomovej práce bolo vytvorenie fyzického modelu z ideového modelu, ktorý bol navrhnutý v programe PRO/ENGINEER. Vytvorenia sa týkalo návrhu na modelu definovať reálne rozmery, spojenia medzi profilmi. Na navrhnutý model bol osadený zotrvačník s hriadeľom do domčekov. Tak isto bolo skompletizované aj prítlačné rameno osadené naviac brzdovým diskom s kotúčovou brzdou.

Po ukončení mechanických prác boli na modeli ďalej osadené kolesá s magnetmi a Hallovými sondami pre snímanie otáčok. Tie nahradili z hľadiska konštrukcie IRC snímače. Roztočením modelu a prevedenou simuláciou modelu sa ukázal čas na zastavenie 0,5 s, a preto bol model doplnený o motor, a tým zabezpečené konštantné podmienky pre akýkoľvek nasadený algoritmus. Motor a brzdu ovládajú RC servomechanizmy pripevnené držiakmi o šasi základni. Dané elektronické komponenty sú ovládané pomocou jednoprocesorovej dosky Arduinou Uno.

Na jednoprocesorovej doske bolo otestované riadenie servomechanizmov pomocou PWM, a správa digitálnych vstupov pomocou tlačidla cez hardvérové prerušenie a cez funkciu. Zvládnutím oživenie jednotlivých prvkov sa prešlo k návrhu algoritmov pre riadenie.

Algoritmy vychádzajú z prvého nápadu, pri ktorom algoritmus fungoval na princípe, kde zabrzdil kolesa a následne začal kontrolovať sklz. Zamyslením sa nad programom dospejeme k tomu, kde algoritmus vedome mal koleso v sklze a nemusel preto vykonávať kontrolu. Z tejto idei preto vyšli algoritmy, ktoré presúvajú výpočet sklzu aj s priemerovacími funkciami potrebnými pre konštantnú hodnotu pulzov do cyklu, v ktorom servo ovláda brzdu. Z dvoch algoritmov testovaných na modeli vyšiel lepšie modifikovaný. Jeho zmenou oproti prvému je použitie cyklu aj pre návrat serva pre novú pozíciu. To sa ukázalo ideálnou voľbou pre zrýchlenie celého cyklu. Pri testovaní boli menené parametre, ktoré sa zadávajú do jednotlivých sklzových možností, tie mali vplyv pri modifikovanom algoritme na namerané výsledky. Parametre pri prvom algoritme mali na zmenu charakteristiky minimálny vplyv. K lepšej ukážke činnosti jednotlivých častí kódu bola k algoritmu vyvinutá aplikácia,

ktorá má za úlohu vizualizáciu a export dát. Aplikácia je písaná v multiplatformovom programe Processing.

Model môže byť základom pre prácu s jednoprocesorovou doskou vlastnou alebo použitou od Arduina. Je možné na ňom meniť algoritmy a tým testovať výsledné časy simulácií.

Do budúcna by som sa zameral na niektoré faktory, ktorými je model limitovaný. Obmedzením pre model sú maximálne otáčky a snímanie pulzov. Riešením by som navrhoval zmeniť motor pre roztáčanie tak, aby dosahoval dvojnásobný počet otáčok 500 ot/s.

Meranie pulzov prebieha za 25 ms. Tento časový úsek je limitou, pri ktorom sa dostáva pre rýchlosť zotrvačníka jeden pulz. Pri testoch serva ovládajúceho brzdu bolo limitným použitím 2 ms, pri ktorom sa stávalo, že koleso ostávalo v sklze. Navrhnutím riešením by boli vložené nosníky pod podpery pre rameno a tým by mohli byť použité IRC snímače. Počet krokov na otáčku sa pohybuje od 500 po 1024, čo by bol oproti terajším 32 pulzov 15 až 30 násobný nárast, ktorým by došlo k skráteniu času.

Zoznam použitej literatúry

30 ROKOV EVOLÚCIE BOSCH [online]. 2008 [cit.2008-1-17] Dostupný z WWW:< http://rb-aa.bosch.com/bosch/infotech/nl-BE/PressText.cfm?mltk=253&tk=254&stk=265>

AIRHELI_SHOP [online]. 2012 [cit. 2013-02-20]. Dostupný z WWW: < http://www.airheli-shop.cz>

Alibaba.com [online]. 2013 [cit. 2013-03-17]. Dostupné z: <http://www.alibaba.com/trade/search?fsb=y&IndexArea=product_en&CatId=&SearchText=sunroof+motor+>

Arduino CookBook. [online]. 2013 [cit. 2013-02-15]. Dostupné z: <http://my.safaribooksonline.com/book/hardware/arduino/9781449321185/4dot-serial-communications/id2562770>

Arduino Examples. [online]. [cit. 2012-10-01]. Dostupné z: <http://arduino.cc/en/Tutorial/HomePage>

Brzdná dráha a jej výpočet. [online]. 2011-01-31 [cit. 2013-04-02]. Dostupné z: <http://www.autorubik.sk/technika/brzdna-draha/>

CAPPA, John. *30 Years Crappy Birthday Bosch ABS* [online]. 2008 [cit. 2013-03-14]. Dostupné z: <http://image.jpmagazine.com/f/editorials/crappy-birthday-bosch-abs/10300701/bosch-abs-history.jpg>

Coefficient of Friction. [online]. 2005-07-28 [cit. 2013-04-01]. Dostupné z: <http://buildingcriteria2.tpub.com/ufc_4_152_01/ufc_4_152_010141.htm>

GME - elektronik [online]. 2012 [cit. 2012-12-20]. Dostupný z WWW: < http://www.gme.sk>

Hallova sonda [online]. 2012 [cit. 2013-04-17]. Dostupné z: <http://www.fpv.umb.sk/~pfeffer/Fyzpraktik/10_uloha_hallova_sonda.pdf>

Inteligentné snímače v automobiloch [online]. 2009 [cit. 2013-03-14]. Dostupné z: <http://isa.own.cz/index.html>

NET Micro Framework [online]. 2012 [cit. 2012-12-20]. Dostupný z WWW: < http://www.prezentace.maxiorel.cz>

NET Micro Framework [online]. 2012 [cit. 2012-12-20]. Dostupný z WWW: < http://informatix.miloush.net/ >

NOSKIEVIČ, P.: *Modelování a identifikace systémů*, Montanex a.s., Ostrava 1999, ISBN 80-7225-030-2

Praktická realizace symetrického stabilizovaného zdroje. [online]. 2011 [cit. 2013-03-20]. Dostupné z: <https://dspace.vutbr.cz/bitstream/handle/11012/6615/Pekarovic_bakalarska_praca.pdf?sequence=1>

Processing. [online]. 2012 [cit. 2013-03-01]. Dostupné z: <http://processing.org/learning/>

Servomechanizmy [online]. 2012 [cit. 2012-12-20]. Dostupný z WWW: < http://www.spslevice.sk/SOC/SOC%20-%20PRI/61-Servomechanizmy.htm>

ŠKODA AUTO a.s., Firemná literatúra, 2004

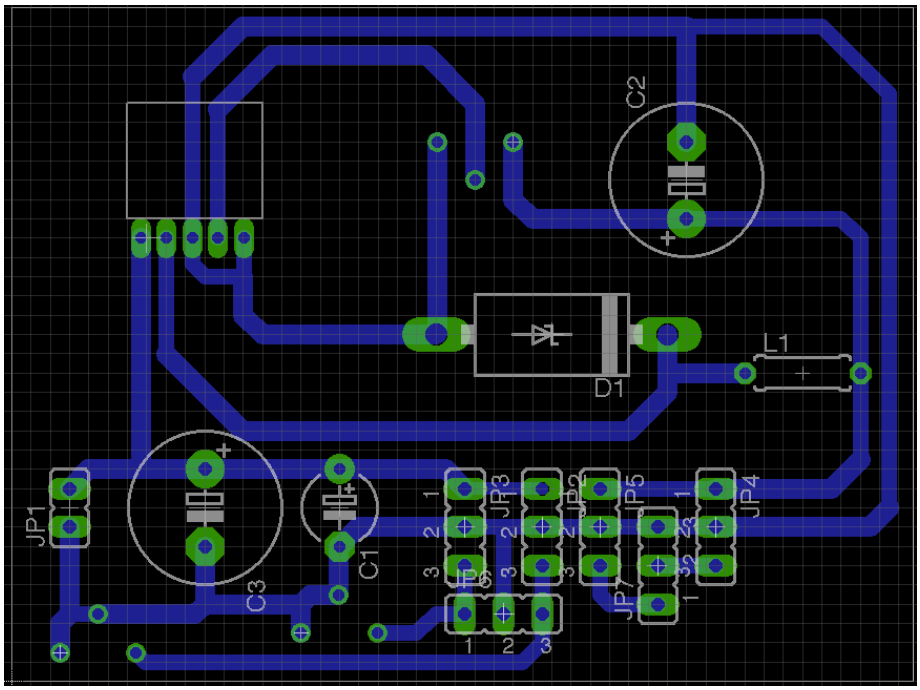
VLK, František.: *Dynamika motorových vozidiel*. 2. vyd. Brno : Nakladateľstvo a vydavateľstvo Vlk, 2000. Obsahuje bibliografiu. ISBN 80-238-5273-6

VLK, František.: *Elektronické systémy motorových vozidel*. 2. vyd. Brno : Nakladateľstvo a vydavateľstvo Vlk, 2002. 2 sv. (298, 299 s.). ISBN 80-238-7282-6

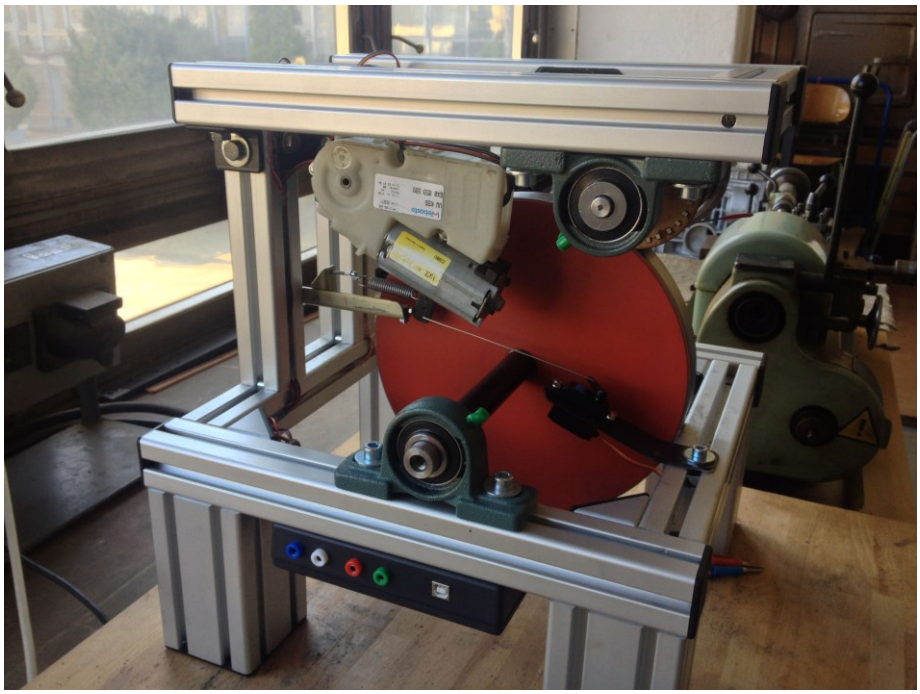
VLK, František.: *Prevodová ustrojí motorovych vozidiel 2*. vyd. Brno : Nakladateľstvo a vydavateľstvo Vlk, 2003. 312 s. Obsahuje bibliografiu. ISBN 80-239-0025-6

Prílohy

Príloha A: Doska plošného spoja v programe EAGLE



Príloha B: Fotka modelu



Príloha C: Podklady pre ďalšiu tvorbu

Podklady z Pro/Engineer

Podklady z AutoCadu

Podklady z Arduina

Podklady z Processingu

Príloha D: Diplom zo STOČ 2013